



Europäisches Patentamt
European Patent Office
Office européen des brevets



(11) EP 0 834 822 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
08.04.1998 Bulletin 1998/15

(51) Int. Cl.⁶: G06F 17/30

(21) Application number: 97307009.7

(22) Date of filing: 10.09.1997

(84) Designated Contracting States:
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC
NL PT SE
Designated Extension States:
AL LT LV RO SI

(30) Priority: 04.10.1996 US 726853

(71) Applicant:
Canon Information Systems, Inc.
Costa Mesa, CA 92626 (US)

(72) Inventors:
• Nehab, Smadar
Costa Mesa, California 92626 (US)
• Wickramaratne, Manjula G.
Costa Mesa, California 92626 (US)
• Klark, Paul L.
Costa Mesa, California 92626 (US)

(74) Representative:
Beresford, Keith Denis Lewis et al
BERESFORD & Co.
2-5 Warwick Court
High Holborn
London WC1R 5DJ (GB)

(54) World wide web news retrieval system

(57) A World Wide Web site data retrieval system includes an input device for inputting data and commands to access the World Wide Web, and a memory for storing a Web site data retrieval driver which includes a Web reader, stored Web site address information, stored Web site commands, and stored format information. The memory also stores process steps to connect to a Web site and to issue commands within the connected Web site, and a connection to the World Wide Web. The system includes a processor for launching the Web site data retrieval driver in response to a command to access the World Wide Web. The Web site retrieval driver, upon being launched, (1) launches the

Web reader to connect to the World Wide Web via the connection, (2) retrieves the Web site address information and Web site commands, (3) instructs the Web reader to access the Web site based on the Web site address information and Web site commands, (4) downloads Web site data from the Web site based on the Web site commands, (5) stores the Web site data in a linear document, (6) repeats steps 1 through 5 until all addresses in the stored Web site address information have been accessed, and (7) formats the linear document into a personalized document based on the format information.

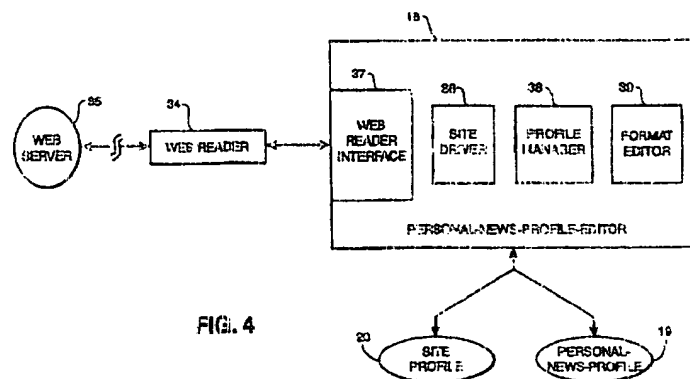


FIG. 4

EP 0 834 822 A2

Description

BACKGROUND OF THE INVENTION5 Field Of The Invention

The invention relates to a data retrieval system which automatically traverses hypermedia documents on a computer network and automatically retrieves information from those documents based on a match between the structure of the documents and a personalized data retrieval structure. More particularly, the invention can retrieve articles from a news service, from a magazine service, or from a combination of both services which are located on the World Wide Web, a private computer network that supports hypermedia links, or any other hypermedia-linked computer system.

For example, there exists a Web site for retrieving news articles from the New York Times and a Web site for retrieving articles from People magazine. The retrieval system of the invention can traverse through such Web sites and select articles based on a personalized data retrieval structure. The personalized data retrieval structure may include commands to retrieve a full text of the front page only, headlines of the business section, headlines of the stock section and sports section, etc. In addition, the personalized data retrieval structure may include content-based rules to retrieve articles with certain keywords, to exclude articles with certain keywords, or to include articles based on a rule-based content analysis. The invention also provides a method for synthesizing all retrieved news articles and printing the synthesized news articles into a newspaper-type format in which each of the articles is arranged based on a user's pre-defined layout.

While the above example is in the context of the Web, hypermedia documents can reside on other types of networks besides the Web, such as an intranet. An intranet is a private computer network that is not connected to outside computer networks. For example, a company's own computer network could be an intranet with hypermedia documents on it. For brevity, the following discussion is made with respect to the World Wide Web. However, it should be understood that the invention applies equally well to any type of computer network that contains hypermedia documents, such as an intranet, different hypermedia-linked computer networks that reside on the Internet other than the Web, etc.

A hypermedia document on the Web can span multiple Web sites. Such documents can be newspapers, news articles, magazines, catalogs, manuals, memoranda, and the like. For brevity, the following discussion is made with respect to sources of news information. However, it should be understood that the invention applies equally well to any other type of hypermedia document.

Description Of The Related Art

The World Wide Web is an on-line source of hypermedia documents containing hypermedia text and images that act as links to other documents, Web sites, etc. As a result, documents on the Web are not organized sequentially. Rather, a user is automatically linked to other documents or Web sites to complete the viewing of a document by selecting a hypermedia link, such as a text link or an image link, within the document. Accordingly, an entire document cannot be viewed by scrolling through text.

One popular use of the Web is on-line publication and distribution of magazines and newspapers. Currently, many Web news services, such as the New York Times, allow the user to define keywords of interest and to receive news information, daily or hourly, that contains text matching the keywords. The news information can then be delivered to the user's computer via modem or E-mail. However, most Web news site newspapers, like the New York Times, include too much information, most of which has no interest to the user since the information is retrieved based only on a keyword match.

Other sources of news information are provided through information suppliers like "Individual Inc." Individual Inc. supplies users with a brief summary of the top twenty most relevant articles based on a user's predefined keywords. This subscription news service allows the user to specify five to ten areas of interest based on keywords, which are then prioritized by the user. The information service searches the Web for magazines and newspapers which contain any of the keywords. Based on the keyword searches, twenty of the most relevant articles are selected, compiled into a brief one-page summary, and transmitted to the user via facsimile for the user's review. However, in order to review an entire document rather than the summary, the user must log onto a specific Web site containing the document in order to retrieve and review the document.

There are yet other services which permit the user to personalize a newspaper to be displayed at the user's terminal by storing links to various news articles from various news sources on the Web. For example, CRAYON "Create Your Own Newspaper" permits a user to select specific sections from among links to over twenty-five different on-line newspapers, and to compose the selections into a personalized newspaper. Using CRAYON, it is possible to compose a personalized newspaper containing, for example, links to the international section of the New York Times, the business section of the Wall Street Journal, and the sports section of the Chicago Tribune. The HTIAL (hypertext markup lan-

guage) source file for this newspaper is then stored to mass media storage for later use.

While the forgoing news and information services provide convenient ways to keep updated on the news, they do not allow a user to access and view the news in the way that people naturally read a real-world newspaper. Namely, people naturally read a newspaper by scanning the pages of sections that they find interesting and then reading those articles that grab their attention. In other words, people use a structural approach to decide what pages to look at initially (e.g., the first page of the Business and World sections, and the comics page of the Arts section). They then scan the selected pages for articles.

In sum, conventional news and information services do not allow a user to access data from a hypermedia document on the basis of the structure of the document, and then to format that data in a manner that allows the user to scan and read the data in a natural fashion.

SUMMARY OF THE INVENTION

In accordance with one aspect the invention addresses the above deficiencies in the art by accessing at least one hypermedia document, retrieving data from the hypermedia document into an extracted data tree, with the data retrieved based on a structure of the hypermedia document, flattening the extracted data tree into a linear document, and formatting the linear document into a formatted document.

In another aspect, the invention creates a personal-news-profile for retrieving data from a hypermedia-linked computer network. The hypermedia-linked computer network is accessed, a learning mode is started, the hypermedia-linked computer network is traversed with commands, at least one rule is extracted from the commands, and the rule(s) is compiled into the personal-news-profile.

In yet another aspect, the invention creates a personalization profile for a Web site retrieval data retrieval system. Data and commands are input to access the World Wide Web and a connection is made to the World Wide Web. A Web reader is launched, and the Web reader accesses the Web via the connection. In response to user commands, a learning mode is entered into. Commands are sent to traverse the World Wide Web, and at least one rule is extracted from the commands. The rule(s) is compiled into a personalization profile, which is stored.

In yet another aspect, the invention retrieves articles from a hypermedia-linked computer network and formats the articles into a personalized newspaper. A stored personal-news-profile is retrieved. The personal-news-profile includes address data for a site on the hypermedia-linked computer network, command data for accessing data from the site, and newspaper layout commands. The site is accessed based on address data stored in the personal-news-profile, and articles at the site are downloaded based on command data stored in the personal-news-profile. The downloaded articles are flattened into a linear document, and the linear document is formatted into the personalized newspaper according to newspaper layout commands stored in the personal-news-profile.

In yet another aspect, the invention retrieves data from a World Wide Web site and formats the data into a personalized document. A Web site data retrieval driver which includes a Web reader, stored Web site address information, stored Web site commands, and stored format information is accessed. The invention (1) launches the Web reader to connect to the World Wide Web via a connection to the Web, (2) retrieves the Web site address information and Web site commands, (3) instructs the Web reader to access the Web site based on the Web site address information and Web site commands, (4) downloads Web site data from the Web site based on the Web site commands, wherein the data is downloaded with reference to a linked list so as to avoid hypermedia-links that form loops and so as to avoid repetitious downloading of data that has already been downloaded, (5) stores the Web site data in a linear document, (6) repeats steps 2 through 5 until all addresses in the stored Web site address information have been accessed, and (7) formats the linear document into the personalized document based on the format information.

In yet another aspect, the invention accesses and retrieves data at World Wide Web sites and formats the data into a personalized document. The invention connects to the World Wide Web, retrieves user defined Web site address information, user defined Web site commands, and user defined formatting commands, and activates a Web reader so as to access a Web site based on the user defined Web site address information. The Web reader is used to download data from the Web based on the user defined Web site commands, and the data is downloaded into an extracted data tree. The downloading continues until all addresses in the user defined Web site address information have been accessed. The extracted data tree is flattened into a linear document, and the flattened document is formatted into the personalized document based on the user defined formatting commands.

In yet another aspect, the invention retrieves news articles from on-line news services on the World Wide Web and formats the news articles into a personalized newspaper. The invention stores a personal-news-profile which comprises addresses data and command data for accessing data from a Web site and newspaper format commands, retrieves the stored personal-news-profile and accesses the data stored therein, activates a Web reader to contact a Web site based on address data stored in the personal-news-profile, downloads news articles at the contacted Web site based on command data stored in the personal-news-profile, stores the downloaded news articles, and formats the stored news articles into the personalized newspaper based on the newspaper format commands stored in the personal-news-profile.

In yet another aspect, the invention formats a hypermedia document into a personalized document. A location of the hypermedia document is specified, a type of the hypermedia document is specified, a scope of data to be retrieved from the hypermedia document is specified, wherein the scope is based on a structure of the hypermedia document, and a format is specified for formatting the data retrieved from the hypermedia document into the personalized document. The hypermedia document found at the specified location is accessed, data is retrieved from the hypermedia document in accordance with the specified hypermedia document type and in accordance with the specified scope, and the data is formatted into the personalized document in accordance with the specified format.

In yet another aspect, the invention is a system for processing a hypermedia document. The system accesses the hypermedia document, extracts addresses from the hypermedia document, and stores the addresses extracted from the hypermedia document in a container. The system activates a processing function to process data stored at the addresses stored in the container, downloads the data stored at the addresses stored in the container into a memory, and extracts predetermined data from downloaded data in accordance with predetermined configuration information. The predetermined data is then formatted in accordance with predefined formatting settings to generate a formatted document, and the formatted document is processed in accordance with the processing function.

In preferred embodiments, the system inputs the formatting settings and configuration information via a graphical user interface. The graphical user interface comprises plural processing icons, one of which activates the processing function. By virtue of the graphical user interface, a user can interactively set a document's format and change that format should a change be desired.

In particularly preferred embodiments, the graphical user interface is displayed in plural modes. The plural modes comprise (1) a fully-functional mode in which the graphical user interface displays formatting fields, processing options, menus and the processing icons, and (2) a minimizing mode in which the graphical user interface displays only the processing icons. Typically, the graphical user interface displayed in the minimizing mode is displayed during browsing the hypermedia document. By displaying the graphical user interface in plural modes, the present invention facilitates operation of the invention during browsing of the hypermedia document.

This summary has been provided so that the nature of the invention may be understood quickly. A more complete understanding of the invention can be obtained by reference to the following detailed description of the preferred embodiments thereof in connection with the attached drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a perspective view showing the outward appearance of the personal news retrieval system according to an embodiment of the invention.

Figure 2 is a block diagram of the personal news retrieval system shown in Figure 1.

Figure 3, comprised of Figures 3A, 3B, 3C and 3D, shows representational diagrams illustrating an example of the transformation of information from the Web (Figure 3A) to an extracted data tree (Figure 3B), then to a flattened document (Figure 3C), and finally to a formatted document (Figure 3D) according to an embodiment of the invention.

Figure 4 is a representational block diagram of the manner by which a personal-news-profile for retrieving news articles via the Web is created or edited according to an embodiment of the invention.

Figure 5, comprised of Figures 5A and 5B, shows flow diagrams describing how a personal news-profile is created or edited.

Figure 6 is a representational block diagram of the manner by which news articles are retrieved from the Web and formatted with reference to a personal-news-profile according to an embodiment of the invention.

Figure 7 is a flow diagram describing how news articles are retrieved from the Web with reference to a personal-news-profile.

Figure 8 is a flow diagram showing how retrieved news articles are formatted with reference to a personal news profile and sent to a print device interface.

Figures 9A to 9E depict a graphical user interface used with the second embodiment of the present invention.

Figure 10 is a flow diagram describing the operation of the second embodiment of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Figure 1 is a view showing the outward appearance of a preferred embodiment of the invention. Shown in Figure 1 is computing equipment 1, such as a Macintosh or an IBM PC or a PC-compatible computer, having a windowing environment, such as Microsoft Windows. Provided with computing equipment 1 is display screen 2, such as a color monitor or a monochromatic monitor, keyboard 3 for entering text data and user commands, and a pointing device such as mouse 4 for pointing and for manipulating objects displayed on display 2. Computing equipment 1 also includes a mass storage device such as disk drive 5. Image data can be input into computing equipment 1 from a variety of sources such as a network interface 11a or from external devices via facsimile/modem interface 6. Network interface

11a is used to connect computing equipment 1 to a local area network (LAN) or to a wide area network (WAN) such as the World Wide Web.

Figure 2 is a detailed block diagram showing the internal construction of computing equipment 1. As shown in Figure 2, computing equipment 1 includes central processing unit (CPU) 8 interfaced with computer bus 9. Also interfaced with computer bus 9 is printer interface 10, fax/modem interface 6, display interface 11, network interface 11a, keyboard interface 12, mouse interface 13, main memory 14, and disk drive 5.

Main memory 14 interfaces with computer bus 9 so as to provide random access memory storage for use by CPU 8 when executing an application such as personal-news-profile editor 16 or Web printer 17. More specifically, CPU 8 loads these software applications from disk drive 5 into main memory 14 and executes the software applications out of main memory 14. In accordance with user instructions, stored application programs are activated which permit processing and manipulation of data. Typically, the software applications stored on disk drive 5, such as personal-news-profile editor 16, Web printer 17, and HTML formatter 18, have been stored on disk drive 5 by downloading the software applications from a computer-readable medium such as a floppy disk or CD ROM, or by downloading the software applications from a computer bulletin board.

Disk drive 5 stores data files which can include text files and image files in compressed or uncompressed format, and stores software application files such as those noted above. The software application files include Windows applications, DOS application, and personal news retrieval files 15. Personal news retrieval files 15 include personal-news-profile editor 16, Web printer 17, HTML formatter 18, personal-news-profile(s) 19, and site profile(s) 20. The detailed functions of personal news retrieval files 15 will be discussed below, after a brief overview of the operation of the personal news retrieval system.

Overview of Document Retrieval

Figure 3, comprised of Figures 3A to 3D, illustrates the operation of a representative embodiment of the invention. Figure 3A is a graphical representation of a typical Web site 21 with news information contained therein. Within Web site 21 is homepage 22 with links to indices such as headings 23, which are in turn linked to articles 24. Some of articles 24 are linked to other articles. As article H 26 resides on another Web site, link 25 is a cross-site link. Link 25 illustrates how a single hypermedia document, represented by homepage 22, can traverse multiple Web sites.

In order to retrieve news from Web site 21, the invention first traverses Web site 21 to retrieve data according to user-defined rules. As will be discussed in more detail below, these rules can be based on the structure of Web site 21, or on the structure of Web site 21 and its contents. The data is retrieved into an extracted data tree, which preserves the organization of the data as shown in Figure 3B, but in which some links are excluded.

The organization of extracted data tree 27 has several features. First, extracted data tree 27 has root 28 which can have child nodes for one or more sites 29, which in turn can have index nodes 30 which correspond to indices/headings 23, articles nodes 31, and the like. Second, extracted data tree 27 is a true tree, with no loops (i.e., cyclic paths) therein. For example, Figure 3A shows a loop from homepage 22 to index node #1, to article C, and then back to homepage 22. This loop is removed when creating extracted data tree 27.

Second, the organization of extracted data tree 27 depends on how the Web sites are traversed, and not on the Web sites' actual layouts. Thus, article H 26 appears under index node #3 (under site #1), indicating that the news retrieval system accessed article H 26 from site #1 via cross-site link 25.

Finally, as noted earlier, certain articles have been excluded from extracted data tree 27 due to the structure of Web site 21 or possibly a content or index/headings 23 and/or articles 24. For example, articles E and G have been excluded from extracted data tree 27.

According to this embodiment of the invention, extracted data tree 27 is flattened into linear document 32, as shown in Figure 3C, possibly with reference to more exclusion rules. Linear document 32 is simply a continuous document with information from extracted data tree 27 embedded therein.

Finally, linear document 32 is formatted according to user specified (or default) formatting instructions into formatted document 33, shown as a stylized personal newspaper in Figure 3D. Formatted document 33 has various fonts and/or colors for site labels, indices/headings, articles, and the like. Furthermore, formatted document 33 is broken down into pages.

Note that in alternate embodiments of the news retrieval system, certain stages of the above transformation from Web site 21 to formatted document 33 can be skipped. For example, data from Web site 21 can be retrieved directly into flattened document 32, as long as a record of the organization of the data is maintained (possibly in a separate linked list) so as to avoid downloading the same article twice, and so as to avoid loops in the organization of Web site 21. Alternatively, extracted data tree 27 can be directly formatted into formatted document 33. In any case, the basic operation of the invention remains the same: the news retrieval system traverses a hypermedia document on the Web, extracts data according to user-defined instructions, and formats the data into a personalized newspaper.

As mentioned in the above discussion, various user-defined rules and other information (such as formatting infor-

mation) are involved in the news retrieval process. That user defined information is stored in personal-news-profile(s) 19, the definition of which is described next.

Defining a Personal-News-Profile

Figures 4 and 5 illustrate the process by which personal-news profile 19 is defined. To create personal-news-profile 19, personal-news-profile editor 16 communicates with personal-news-profile 19, site profile 20, and Web reader 34.

Personal-news-profile 19 contains information as to what sites to access for creating a personalized newspaper, what sections to retrieve from those sites, rules to be used to determine what data to extract from the sections and the articles therein, rules to determine how to exclude links, and newspaper format information. A sample personal-news-profile is shown in Appendix 1.

Site profile 20 includes general site information that is not specific to a particular user. For example, site profile 20 could contain information such as full site addresses, sections within a site, non-user specific passwords, etc. Sample site profiles are shown in Appendix 1. Because general site information is stored in site profile 20, personal-news-profile 19 can refer to the general site information with reference to site profile 20, saving space in the personal-news-profile. For example, as shown in Appendix 1, personal-news-profile 19 can refer to a site number 1. Site profile 20 indicates that site number 1 is the "San Jose Mercury News," with a homepage at "http://www.sjmercury.com/". This construction also centralizes general site information. Thus, if a site address changes, only site profile 20 needs to be changed to update all personal-news-profiles 19 on the system.

Web reader 34 is an application program or program module that communicates with the Web via Web server 35. In response to commands from personal-news-profile editor 16, Web reader 34 will access the Web, traverse hypermedia documents on the Web, retrieve data from the documents, and return the retrieved data to personal-news-profile editor 16.

As shown in Figure 4, personal-news profile editor 16 includes four modules: site driver 36, Web reader interface 37, profile manager 38, and format editor 39.

Web reader interface 37 interfaces personal-news-profile editor 16 to Web reader 34. Site driver 36 interacts with Web reader 34 via Web reader interface 37 to provide an abstract interface to each individual Web site. More specifically, site driver 36 instructs Web reader 34 to access various Web sites and to retrieve data from those sites. Thereafter, site driver 36 receives that data and builds site profile 20 therefrom. The data may also be used to update an existing site profile.

In building site profile 20, site driver 36 translates the structure of each accessed Web site to a uniform structure defined in site profile 20, and stores data retrieved therefrom in site profile 20. By translating different Web sites, some of which may have different structures, into a single uniform structure, and storing data therefrom in that structure in site profile 20, the present invention facilitates access to information from different Web sites, and thus reduces overall processing time.

Profile manager 38 maintains document templates that specify how to format a personalized newspaper. Predefined document templates exist. In addition, format editor 39 allows a user to specify personalized templates for formatting a newspaper, either by editing existing templates or by creating new ones. In any case, each document template specifies page layout information, font information, style information, colors, etc. for the titles, indices/headings, sub-headings, text and the like for a personalized newspaper.

Sample code for personal-news-profile editor 16, site driver 36, and profile manager 38 is included in Appendix 3A.

Figures 5A and 5B are flow diagrams describing the operation of personal-news-profile editor 16 in more detail. Figure 5A shows the operation of personal-news-profile editor 16 in defining the parts of personal-news-profile 19 relating to accessing Web sites and retrieving data from those sites.

In step S500 of Figure 5A, personal-news-profile editor 16 is launched by a user. In step S501, the editor launches Web reader 34. The user's personal I.D. is then retrieved from a file. If a personal-news-profile already exists for that I.D., step S503 directs flow to step S504, where the user is given the option of skipping to the format editor. Otherwise, personal-news-profile editor 16 enters a "learning mode" in step S505. Once in the learning mode, personal-news-profile editor 16 proceeds to step S506, where it accepts a Web command (i.e., a command to traverse a hypermedia link) from the user and forwards the Web command to the Web reader 34 in terms of site profile 20. Site driver 36 maintains a hierarchical log of Web sites visited by Web reader 34. In step S507, personal-news-profile editor 16 creates an extraction rule from the Web command. This rule will allow the hierarchical system to later duplicate the user's selection criteria in browsing (clicking on hyperlinks) visiting a Web site.

The rule specifies, at the least, structural criteria for duplicating the traversal of the Web site. For example, if a user accesses all articles under a particular index/heading, the rule will specify that all articles under that index/heading should be retrieved.

In one embodiment of the invention, the rule can also include content-based criteria (i.e., keyword-based criteria) accepted from the user. These content-based criteria can, for example, require a certain word to be in an article, (2)

exclude articles with certain words, (2) exclude certain boolean combinations of words, (3) rank articles that are selected based on structural criteria, with the ranking based on keywords, and then require the selection of the articles with the highest ranking(s), or (5) exclude certain types of articles such as advertisements.

Examples of the syntax for the structural and content based exclusion rules are shown in Appendix 2. Several different types of rules are shown. Some simply limit the traversal of a Web site to a certain number of links. Others are date and keyword based exclusion rules. One particularly useful rule indicates that articles should be ranked based on a keyword analysis and the top scoring articles should be chosen. Other rules include "flattening" rules. These rules control the flattening of the extracted data tree, as will be explained in more detail below.

At the least, the rule includes structural information about the user's selection (i.e., first page, first document, all links, etc.), necessary password information, browser commands, and the like. The rule can also include a pointer or a reference to site profile 20 and the appropriate information therein. General (non-user specific) information is used by site driver 36 to maintain site profile 20. In this manner, address information and passwords common to multiple users can be maintained in site profile 20, as discussed above. For example, site driver 36 will store commands or hyperlinks to other documents in a Web page in the rule, but will not store a Web site's full address in the rule. That address information is stored in site profile 20.

In step S508, rule data defining the rule created from a Web command(s) is stored in an extracted data tree such as extracted data tree 27 in Figure 3b. This data tree is a linked list that reflects the organization of the data retrieved from the Web. In step S509, flow returns to step S505 for the next Web command unless the user is done (i.e., the user signs-off the Web site), in which case flow proceeds to step S510.

At this point, the creation of the personal-news-profile has proceeded much like the creation of a macro common to word processing programs, except that site profile 20 has been used to minimize storage requirements and to centralize general site information. In order to minimize storage requirements further and in order to make the news retrieval system more flexible and efficient, the extracted rules are now compiled to remove redundant links, multiple visits to the same site, and the like. This occurs in step S510, and the resulting compiled rules become the first part of personal-news-profile 19.

Alternatively, personal-news-profile editor 16 may be involved as a graphical user interface which allows a user to edit a previously stored personal-news-profile or to specify document composition preferences, for example, by specifying news sites, headline articles only, keywords, etc. In either case, the result is personal-news-profile 19, which comprises a listing of Web site pointers as well as extracted rules for traversing through a Web site or sites.

For a better understanding of the above, sample personal-news-profiles and sample site profiles are provided in Appendix 1 as noted above.

Next, operation proceeds to give the user an option to copy a custom newspaper template, as shown in Figure 5b. In step S511, it is determined if a newspaper template has been defined and stored in personal-news-profile 19. If a newspaper template has been defined, step S512 gives the user the option to edit the template or to proceed to step S520. If the user chooses to edit the template or if no newspaper template has been defined, flow proceeds to step S513.

Step S513 gives the user the option of creating a custom template or using a predefined template. If the user wants to use a predefined template, step S514 gets the specified predefined template which is added to the personal-news-profile in step S519. Otherwise, flow proceeds to step S515 where format editor 39 is involved.

Format editor 39 has a graphical user interface that provides the user with a number of formatting options. In step S516, format editor 39 allows the user to define which newspaper sections are to be printed in the newspaper, which Web site's news articles are to be placed in each section, and how each page is to be laid out. In this regard, the user can specify which Web site's news articles are to be used as a business page, which Web site's news articles are to be used as a sports page, etc. In addition, in step S516, the user can define where each individual heading should be laid out, to what its attributes (headings should go on each page.

In step S517, format editor 39 allows the user to edit the newspaper's layout headings, sub-headings, bylines and actual text of news articles. In step S518, format editor 39 permits the user to define index/heading colors, title colors, etc. In this regard, layout editor 39 is capable of displaying the types of fonts and colors available to the user based on the system's printer capabilities.

Once all of the information is gathered from the user and used, the format editor 39 adds the information to personal-news-profile 19 in step S519. Alternatively, profile manager 17 may also store the custom format as a template in a common area for use by other users. In this case, only a copy of the reference information to template is stored in personal-news-profile 19.

In step S520, personal-news-profile editor 16 prompts the user to set an alternate newspaper delivery time and method (i.e., print or store on disk for later retrieving). These settings are added to personal-news-profile 19. More specifically, in the case that a user's computer is continuously supplied with power, the news retrieval system can be launched automatically at a designated time. The system will then retrieve from the Web sites which are listed in personal-news-profile 19. Upon retrieving the information, the articles will be organized based on the newspaper tem-

plate in personal-news-profile 19. The formatted personalized newspaper can then be either printed or stored for later viewing. In the case that a time is not set for newspaper delivery, the user can execute the Web news retrieval system program at any time.

Once personal-news-profile 19 has been created, the Web news retrieval system, upon being launched, can traverse Web news sites and build a personalized newspaper by automatically retrieving various news articles from the Web news sites and print the news articles based on the newspaper template indicated in personal-news-profile 19. A description of how the Web news retrieval system of the invention performs this function is described next.

Retrieving a Document Using a Personal-News-Profile

Figure 6 is a representational block diagram of the manner by which the invention retrieves articles from the Web according to personal-news-profile 19. (Figure 6 also shows in a manner by which the retrieved articles are flattened into a linear document and formatted. These functions are discussed in greater detail in the next section of this application.)

As shown in Figure 6, Web printer 17 is responsible for retrieving news articles. Web printer 17 is an end-user application that communicates with personal-news-profile(s) 19, site profile 20, Web reader 34, and output interface 40 in order to perform this function.

Web printer 17 looks at personal-news-profile 19 to determine which Web sites to access and which data to retrieve from those sites. Web printer 17 also looks at site profile 20 for general site information. According to the information in personal-news-profile 19 and site profile 20, Web printer 17 insures Web reader 34 is connected to the Web via Web server 35 in order to access various Web sites and to retrieve data from those sites. Web reader 34 sends the retrieved data to Web printer 17, and Web printer 17 uses the data to build an extracted data tree. As will be discussed in greater detail in the next section of the application, Web printer 17 then flattens the extracted data tree into a linear document and formats the linear document for output via output interface 40.

As shown in Figure 6, Web printer 17 includes four program modules: Web reader interface 50, site driver 51, tree manager 41, and formatter 42.

Web reader interface 50, like Web reader interface 37 described above, interfaces Web printer 17 to Web reader 34.

Site driver 51 accesses site profile 20 and personal-news-profile 19 and provides data stored therein to Web reader 34. As noted above, Web reader 34 uses that data to access various Web sites and to extract data therefrom. As noted above, this retrieved data is used by Web printer 17 to build an extracted data tree.

Tree manager 41 manages the extracted data tree. In the regard, tree manager 41 keeps track of the organization of the retrieved data in the extracted data tree. This allows Web printer 17 to avoid accessing the same article twice, to avoid unnecessarily re-visiting a Web site, and to avoid getting caught in a cycle (loop) in the organization of a hyper-media document on the Web. Alternatively, tree manager 41 could store the data in blocks (as opposed to directly in a data tree) with reference to a linked list that provides the same functionality as the extracted data tree. Sample code for tree manager 41 is included in Appendix 3B.

Formatter 42 is responsible for flattening the extracted data tree into a linear document and formatting the linear document into a personalized newspaper. Formatter 42 performs these functions in accordance with the print criteria and format information (i.e., newspaper template) indicated in personal-news-profile 19. Sample code for formatter 42 is included in Appendix 3B.

In more detail, Figure 7 is a flow diagram describing how Web printer 17 uses Web reader 34 to traverse the Web according to personal-news-profile 19 and to retrieve articles from the Web according to the profile, excluding unwanted data.

The Web printer starts in step S701. In step S701, Web printer 17 retrieves either the designated personal-news-profile or a default personal-news-profile stored in its storage using unit 44. In this regard, because computer equipment 1 may be used by more than one user, there may be one or more personal-news-profiles stored on the equipment, one of which will be designated as the default. Upon retrieving the designated personal-news-profile, in step S702 Web printer 17 determines whether any news data has been previously stored to disk drive 5 (for example, by a previous automatic news delivery), or if new articles should be retrieved using personal-news-profile 19.

In the case that news data does exist on disk drive 5, in step S703 the stored news data is retrieved and flow proceeds to step S801 of Figure 3, discussed in more detail in the next section. On the other hand, if no stored news data exists, Web printer 17 invokes Web reader 34 in step S704 to find out what data is stored in Web reader 34 as discussed above with respect to defining a personal-news-profile.

Upon being invoked, Web reader 34 connects to Web server 35 in step S705, if Web printer 17 is a client, or to a network, such as the World Wide Web. Web printer 17 then provides Web reader 34 with information for the first Web site to be visited based on information retrieved from personal-news-profile 19. Once connected to the desired Web site in step S706, Web printer 17 provides Web reader 34 with a URL link for retrieving a link to the next Web page containing information that personal-news-profile 19 indicates should be retrieved. Web reader 34 traverses the Web

according to this information in step S707.

In step S708, Web reader 34 retrieves the desired information and sends it to Web printer 17 according to the rules in personal-news-profile 19. Thus, data exclusion occurs in this step. The rules in personal-news-profile 19 specify structural and content-based criteria for excluding data from the personalized newspaper. The structural rules limit the
 5 retrieved information on the basis of the structure of the Web site accessed by Web reader 34. The content-based rules limit the retrieved information on the basis of its content. As mentioned above, in step S709, in creating a personal-news-profile, examples of the syntax of the retrieval rules in personal-news-profile 19 are included in Appendix 2.

In addition to rule-based exclusion, media-type exclusion occurs in step S708, wherein data of a media type that can not be printed is excluded from the extracted data tree. For example, movie and sound data can be excluded.

10 Web printer 17 stores the retrieved data in disk drive 5 (or in main memory 14) in the extracted data tree managed by tree manager 41. Alternatively, the data could be stored in blocks with reference to a linked list, as discussed earlier. In step S709, Web printer 17 returns to step S707 to complete retrieving all information from Web pages at the Web site. In step S710, upon completing a traversal of one Web site, Web printer 17 uses tree manager 41 to compare the
 15 sites remaining in personal-news-profile 19 with the site organization information in the extracted data tree to determine if more sites need to be visited. In the case that more Web sites need to be visited, step S710 returns flow to step S706 and news articles are retrieved in the same manner as discussed above. On the other hand, if all of the Web sites listed in personal-news-profile 19 have been visited and all of the articles retrieved, flow proceeds to step S801 in Figure 8.

Flattening and Formatting the Retrieved Data

20 Figure 8 is a flow diagram showing how the extracted data tree is flattened and formatted. The configuration of the invention is the same as when retrieving data from the Web (shown in Figure 5). In fact, the flattening and formatting processes can occur, at least to a limited extent, concurrently with the data retrieval process.

In step S801 of Figure 8, the extracted data tree is flattened. This simply means that the organization of the data is
 25 converted from an extracted data tree to a linear document. This step provides the opportunity for excluding more data from the personalized newspaper, for example by only including nodes of the data tree into the flattened document. This exclusion process is controlled by the flattening rules in personal-news-profile 19.

After the data is flattened into a linear document, the data is formatted in step S802 according to the template indicated in personal-news-profile 19. The definition of this template, which is either a predefined template or a custom
 30 template, was discussed earlier. Finally, in step S803, the formatted and fully personalized newspaper is sent to output interface 40. This interface could be printer interface 10, printer 7, display interface 11 to display 2, or even modem/fax interface 6.

Second Embodiment: The HTML Formatter

35 The second embodiment of the invention is a system for processing a hypermedia document. The system accesses the hypermedia document, extracts addresses from the hypermedia document, and stores the addresses extracted from the hypermedia document in a container. The system activates a processing function to process data stored at the addresses stored in the container, download the data stored at the addresses stored in the container into
 40 a memory, and extracts predetermined data from downloaded data in accordance with predetermined configuration information. The predetermined data is then formatted in accordance with predefined formatting settings to generate a formatted document, and the formatted document is processed in accordance with the processing function.

The second embodiment of the invention is exemplified as HTML formatter 18, shown in Figure 8. An example of HTML
 45 formatter 18 is WebFormatter, manufactured by Data Information Systems, Inc. The second embodiment will be described with respect to WebFormatter. It should be noted, however, that HTML formatter 18 is not limited to the WebFormatter embodiment, and that various alternative embodiments within the spirit and scope of the following description are possible.

WebFormatter is stand-alone utility software that can be used in conjunction with different Web browsers, such as Netscape, Mosaic and Internet Explorer. In short, WebFormatter extracts data from a Web page, strips out extempora-
 50 neous data from the extracted data, and reformats the data into a formatted document. The formatted document can then be printed, stored in an RTF (Rich Text Format) file or edited in any RTF-compatible editor, such as MS Word, WordPerfect, Wordpad, etc.

WebFormatter can be activated from a windowing environment such as Microsoft Windows[®]. From such a windowing environment, WebFormatter can be activated by double-clicking on a WebFormatter icon (not shown) in a start-up window, selecting WebFormatter from the Windows Start menu, dragging a WebFormatter icon (not shown) from a Web browser and dropping it on the WebFormatter icon in the Start menu, or double-clicking WebFormatter
 55 when the Web browser is started.

Unlike the first embodiment of the invention described above, WebFormatter does not use a predefined personal-

news-profile to specify criteria for creating a particular type of document from one or more Web pages. Rather, WebFormatter relies upon user-specified criteria to create a particular type of document, such as a newspaper or the like, from one or more Web pages. These criteria are input interactively by a user via a graphical user interface.

As described in more detail below, WebFormatter operates in two modes - a minimized mode and a fully-functional mode. In the minimized mode, WebFormatter's graphical user interface is essentially a "push to print" button, which is displayed concurrently with displayed Web pages. By virtue of this feature, as a user "surfs the Web" the user can process, format, and print out Web pages by merely clicking on the "push to print" button.

In its fully-functional mode, WebFormatter's graphical user interface provides spaces for a user to enter a URL address of a Web page to be processed, enter a personal title for the document, select a format for the document, preview a formatted first page of the document, and either print the document, save the document as an RTF file, or view/edit the document using an RTF editor. The graphical user interface for the fully-functional mode will be described first, since it is from that interface that the user can enter the minimized mode.

Figure 9A shows graphical user interface 43 for WebFormatter's fully-functional mode. Graphical user interface 43 is displayed on display 2 upon first activation of WebFormatter. As with any interactive windowing software application, a user interacts with graphical user interface 43 by means of mouse 4 (by pointing and clicking) and keyboard 3.

As shown in Figure 9A, graphical user interface 43 includes fields 44 and 46 to 49, through which a user can specify the URL address of a document to be formatted and the format of that document. Beginning with URL field 44, a user enters the URL address (e.g., <http://www.sas.com/condit/legis/watwin/>) of a Web page to be processed by WebFormatter. There are several different ways for the user to enter a URL address. The user can: (1) type the address directly into URL field 44, (2) copy the URL address from the Web browser and paste the URL address into URL field 44, (3) drag the URL address from the Web browser onto graphical user interface 43 or onto the WebFormatter icon, or (4) click on Current URL button 54.

With regard to Current URL button 54, if a user clicks on Current URL button 54, WebFormatter locates the active Web browser and queries the Web browser for the address of the current Web page. Thereafter, the Web browser provides the address of the current Web page to WebFormatter, which places the address in URL address field 44. If URL button 54 is activated and no Web browser is currently running, WebFormatter displays dialog box 56, shown in Figure 9A.

As shown, dialog box 56 includes Cancel button 57 and Launch Browser button 59. Cancel button 57 cancels a user's request to input a URL address into URL address field 44. Current URL button 54, Launch Browser button 59, on the other hand, launches a Web browser specified in WebFormatter. As noted below, WebFormatter is configured beforehand with predefined information including a Web browser to be used with WebFormatter. Configuration of WebFormatter will be described in more detail below.

In alternative embodiments of WebFormatter, a file name can also be entered into URL address field 44. For example, in these alternative embodiments, if a user wishes to format a hyperlinked file that has a book-like format, the user enters the filename into URL address field 44. Thereafter, WebFormatter proceeds through the file in the same manner as through specified Web pages in order to reformat the hyperlinked material as desired.

Returning to graphical user interface 43, title field 46 prompts the user to enter a personal title for a formatted document. The title may be typed directly or pasted into title field 46.

Formatting fields 47 to 49 define the format of a document to be output by WebFormatter. Options for the different formatting fields can be accessed by clicking on a scroll bar, such as scroll bar 53, of a respective formatting field. Each of these fields is described in detail below.

Styles field 47 provides four options for formatting a formatted document. These options relate to characteristics of an output document such as size of output, margins, etc. The four options include Condensed, Easy To Read, Formal, and Professional. The invention of course is not limited to these four options; any other options may be added as desired.

Columns field 48 defines the number of columns in a formatted output document. Two columns options are available - Single and Multiple; however, the invention is not limited to these two options. The Single option, as might be expected, formats the document into a single column. The Multiple option, on the other hand, formats the document into a predetermined number of columns. In preferred embodiments of the invention, the Multiple option is set to two columns; however, any number can be set.

Spacing field 49 defines the spacing between lines of text in a formatted output document. Three options are provided in WebFormatter, but other options can be added as desired. The three options are Condensed, Normal and Easy To Read, with Condensed being the least amount of spacing between lines and Easy To Read being the most amount of spacing between lines.

Graphical user interface 43 is also provided with Preview button 60. By clicking on preview button 60, a user can preview a first page of a formatted document before printing it. An example of a page of a formatted document is shown in Figure 9A.

As shown in Figures 9A and 9B, WebFormatter also includes Options button 61. Options button 61 provides a user with additional formatting options that are used by WebFormatter to create a formatted document. A user can activate

Options button 61 by clicking thereon. This causes Options dialog box 62, shown in Figure 9B, to appear on display 2.

As shown in Figure 9B, options dialog box 62 includes General options 64, Container options 66 and Strip Meta Info options 67. General options 64 includes "Text only" listbox 72, "Index of links in the page" listbox 73, and "No floating pictures" listbox 74. These options are indicated by a check mark for those in a respective listbox. As will become clear from their descriptions, more than one of the options in General options 64 can be selected at the same time.

"Text only" listbox 72 instructs WebFormatter to strip all graphics from Web pages and print only text therein. "Index of links in the page" listbox 73 instructs WebFormatter to add the list of all URLs present in a Web page or pages to the end of a formatted document. Preferably, the list of URLs is printed as superscript, and the positions of the URLs in the list are marked in bold. "No floating pictures" listbox 74 instructs WebFormatter to print all images in the document in a particular area of the formatted document. In some cases, therefore, when this option is selected, WebFormatter shrinks images, as needed, so that images fit into a particular area.

Strip Meta Info options 67 provides engineering options which facilitate stripping of unnecessary information from a Web page being processed by WebFormatter. The options include (1) "None", which instructs WebFormatter to strip nothing from the Web page, (2) "Till the first horizontal line", which instructs WebFormatter to strip all links and images until and up to predefined first and second horizontal marking rules (e.g., up to the horizontal line across a page), and (3) "Till the first text", which instructs WebFormatter to strip all links and images up to first and last occurrences of text in the Web page. Only one of Strip Meta Info options can be selected at a time. Selection thereof is indicated by a dot in a bullet located next to an option, as shown in Figure 9B.

Container options 66 provides options for processing of a format, addresses for which are stored in container 76 shown in Figure 9B. Prior to describing Container options 66, a description of container 76 will be provided.

As noted, container 76 stores URL addresses of selected documents. Document addresses which are input to field 44 are added to container 76. The order in which URLs are input into container 76 denotes the order in which data in the URLs is processed by WebFormatter. As shown in Figure 9B, once container 76 becomes full, its icon changes to that shown by reference numeral 77.

When a user clicks on the icon for container 77, menu 77 is displayed. Menu 77 provides five options; i.e., Open 79, Empty 80, Print 81, Edit 82 and Save 83. These options are highlighted when activated, and are described in detail below.

Open 79, when activated, displays Container Contents screen 87 shown in Figure 9B. Container Contents screen 87 shows the URL addresses stored in container 76. Container contents screen 87 provides four options; i.e., Add current URL button 88 which adds the current URL to container 76, Select button 89 which permits a user to highlight and delete a URL in container 76, Empty button 90 which permits a user to empty container 76, and Done button 91 which permits a user to close Container Contents screen 87. A user may also empty the contents of container 76 by clicking on Empty 80 of menu 77.

In addition, the user can rearrange the order of URLs stored in container 76 by dragging and dropping different URLs at different locations therein. As noted above, since the URLs are processed in the order that they appear in container 76, this feature permits a user to rearrange the processing order of the URLs in container 76 interactively.

Print 81, Edit 82 and Save 83, when activated, cause WebFormatter to download all data at Web pages defined by the URLs stored in container 76, format them as specified by the user, create RTF files storing the formatted Web pages, and do the selected action, i.e., save, edit or print the RTF files. This procedure is described in greater detail below.

Referring back to Options dialog box 62, Container options 66 include "Print contents of contents" listbox 92 and "Empty after processing" listbox 93. As shown, a check mark appears in a listbox to indicate that the listbox has been selected. In this regard, more than one listbox can be selected at a time. The value of contents listbox 92, when selected, instructs WebFormatter to print the contents of container 76 as a series of documents in a formatted output document. "Empty after processing" listbox 93, when activated, instructs WebFormatter automatically to empty container 76 after printing, editing or saving a document, without waiting for a user to do so.

Also shown as part of Container options 66 are Select RTF Editor button 68, Cancel button 69 and OK button 71. By clicking on Select RTF Editor button 68, a user can save an RTF file editor, examples of which are noted above. This can be done, for example, by displaying another dialog box using predefined RTF editors (not shown) and selecting one of the predefined RTF editors. Cancel button 69 causes Container options 66 and OK button 71 confirms selected options in Container options 66 and then returns processing to the user.

As shown in Figure 9C, graphical user interface 94 includes a program icon 95, an icon to save icon 99, help button 100, done button 101 and minimizing icon 102. A user may select any of these features by clicking thereon using a mouse.

Print icon 95 opens a print dialog box 103 shown in Figure 9C, which is used to print any number of copies of Web pages formatted by WebFormatter. Edit icon 97 opens an RTF file editor 104 for editing formatted RTF pages, or editing by a predetermined RTF editor. Save icon 99 opens a save dialog box 105 shown in Figure 9C, which prompts the user to name and save a for-

matted Web page as an RTF file. Help button 100 provides help information for operating WebFormatter, and Done button 101 exits from WebFormatter. Similarly, the "Done" button that exits from the window of WebFormatter which was mentioned above and which is described in greater detail below.

Figure 9C shows menus provided by WebFormatter during its operation. These menus include file menu 103, edit menu 104 and window menu 106. File menu 103 includes "New" option 105, "Open" option 107, "Save" option 109, "Print" option 110, "Quit" option 111, and "Web browser" option 112, all of which are identical to those of Save icon 99, Edit icon 97, and Print icon 98, respectively. An "Exit" option is also provided to exit from File menu 103. Finally, File menu 103 includes "Open HTML file" option 107. This option provides a user with the capability to open a local HTML file, a hypertext file resident on the user's computer, or as a file saved from NetScape, or URL files created by dragging and dropping a URL into the window's desktop. "Open HTML file" option 107 also provides hooks needed to open files created by other Web file-processing products so that those files can be formatted as RTF files and printed, saved and/or edited using WebFormatter.

Edit menu 104 provides "Paste URL" option 108. "Paste URL" option 108 pastes the contents of a paste buffer, such as a URL address copied from a Web page, into URL field 44 as described above.

Window menu 106 provides a "Help Topics" option 113, which provides a user with information regarding the use, maintenance and background of WebFormatter, and an "About WebFormatter" option 114, which provides a user with a dialog box (not shown) containing WebFormatter's version number and copyright notice(s). Window menu 106 also includes "Preferences" option 116. "Preferences" option 116 opens preferences dialog box 112 shown in Figure 9D.

Preferences dialog box 112 is used to configure and reconfigure WebFormatter. As shown in Figure 9D, preferences dialog box 112 includes Minimize view options 115, General options 117, and WWW Browser to use options 118.

Minimize view options 115 can be used to configure WebFormatter's graphical user interface in the minimized mode. Two sets of options are provided. The first set includes "Window", "Full Screen" and "Save". These options correspond to print icon 96, edit icon 97 and save icon 99, shown in Figure 9B. When a checkbox next to one of these options, the icon for that option is displayed in the minimized mode. For example, the print icon, the edit icon and/or the save icon. More than one option can be selected at once. In this regard, Figure 9E shows graphical user interface 116, which is a representative example of a graphical user interface for WebFormatter when WebFormatter is in the minimized mode.

Referring back to Figure 9D, Minimize view options 115 also include "Row" and "Stack" options. These options can be set to display WebFormatter's graphical user interface in the minimized mode horizontally by selecting "Row" or vertically by selecting "Stack". Only one of these options can be selected at a time. An example of the foregoing graphical user interface 116 corresponds to a row of icons.

WWW Browser to use options 118 determine what Web browser is used with WebFormatter. As shown, preferably Netscape, Internet Explorer, and Microsoft's Internet Explorer are typical; however, other browser options can also be provided. As might be expected, only one of these options can be selected at a time. The default browser option is Netscape Navigator.

General options 117 include "Warn before printing more than _____ pages" option 119, "Open in minimized view" option 113, "Warn before printing more than _____ pages" option 120, "Auto-start with browser" option 117, and "WebFormatter to be activated automatically when a Web browser is activated". If this option is not selected (which is the default), WebFormatter is activated by double clicking on a WebFormatter icon in the windowing environment, selecting WebFormatter from the Window's menu, or dragging and dropping a URL from the Web browser into the WebFormatter icon, as described in more detail above. "Open in minimized view" option 118, when selected, opens WebFormatter in the minimized mode. In the minimized mode, WebFormatter is in a full-screen mode. "Warn before printing more than _____ pages" option 119 and "Warn before printing more than _____ pages" option 120 allow a user to control the number of pages saved of a formatted document and the amount of memory space used by those pages, respectively. The default for both of these options is one. Only one of these options can be selected at the same time.

Preferences dialog box 112 also includes Cancel button 121, which cancels a user's selected preferences and OK button 122 which confirms a user's selected preferences.

As explained above, WebFormatter can be configured to enter directly into the minimized mode via Preferences dialog box 112, or a user can enter the minimized mode via minimizing icon 102 shown in Figure 9B. As also noted above, Figure 9E shows an example of graphical user interface 116 for WebFormatter in the minimized mode. Graphical user interface 116 is displayed as a floating interface when a user is exploring the Web. Thus, as a user views a Web page, the user also views graphical user interface 116. When an application icon on graphical user interface 116 (which, in Figure 9E, includes icons for window, full screen, and save) is selected, the user can capture the current Web page, save it as a formatted file, print it, and save, edit and/or print the RTF file. Alternatively, the user can drag a URL from the browser and drop it onto one of the icons.

A user can reconfigure WebFormatter and WebFormatter's graphical user interface by double clicking on the minimize button. This action causes a preferences dialog box to appear on display, which is identical to preferences dialog box 112. Thereafter, the user can alter the configuration of WebFormatter as desired. Should a user wish to enter the full-functional mode from the minimizing mode, the user need merely double click on icon 117 shown in Figure 9E.

Figure 10 is a flow diagram describing the operation of WebFormatter. WebFormatter is activated in step S1000. As described above, this can be done by double clicking on WebFormatter icon in a windowing environment. Depending upon how WebFormatter has been configured, e.g., in the fully functional mode or the minimizing mode, either a graphical user interface similar to that of graphical user interface 43 or one similar to that of graphical user interface 116 is displayed in step S1001. For the sake of convenience, the following description of a graphical user interface similar to that of graphical user interface 43 is displayed in step S1001, since the default mode of WebFormatter is the fully-functional mode.

Next, in step S1001, WebFormatter is configured, as described above via preferences dialog box 112 and options dialog box 62. This step is not necessary unless a user wishes to change WebFormatter's previously set configuration. In step S1002, document format data is input in fields 44 and 46 to 49 described above. More specifically, the user inputs a URL (or filename in alternative embodiments) into URL field 44. As described below, WebFormatter uses this information to process Web pages stored at the URL to create an RTF file based on the configuration of WebFormatter and the data input in fields 46 to 49.

In step S1003, a Web reader similar to that of Web reader 34 described above is executed. The Web reader connects to a network, such as the Internet, via Web, in step S1003.

Next in step S1003, it is determined whether a URL or a filename has been entered. As described above, in preferred embodiments of WebFormatter, only a URL may be entered. However, since alternative embodiments of WebFormatter may permit entry of a filename, a description of processing a file other than one at a URL address will be provided.

If a URL has been entered in field 44, processing proceeds to step S1006. In step S1006, the Web reader accesses the hypermedia document (e.g., a homepage) specified by the URL address. In step S1007, WebFormatter instructs the Web reader to traverse the hypermedia document. Thereafter, WebFormatter extracts URL address(es) from the Web and stores the addresses in container 75. Once all desired addresses have been selected and a processing function, such as print, has been activated, WebFormatter downloads data stored at the addresses in container 76 into memory 5. WebFormatter then extracts predetermined data from the downloaded data based on the configuration information set in Options dialog box 62, and stores the extracted data in memory 5. Thus, for example, if "Text Only" option 72 in Options Window 62 is on, only text is extracted from the downloaded data. Processing then proceeds to step S1011.

On the other hand, if, in step S1003, a filename for an HTML source file is entered, WebFormatter instructs the Web reader to access a first site in the file. In steps S1006 and S1007, the site is traversed and data is extracted and stored in the same manner as in step S1007, described above. Then, in step S1008, WebFormatter determines if more sites are listed in the HTML source file. If more sites are listed in the file, flow returns to step S1008, and the next site is accessed. If no more sites are present, processing proceeds to step S1011.

In step S1011, WebFormatter processes the extracted data in accordance with the previously set format information. For example, if Columns field 46 is set to multiple, the extracted data will be formatted into a document having multiple columns. The above processing is initiated by activating one of Print icon 86, EMail icon 87 or Save icon 99, and is similar to the processing described above in the first embodiment, e.g., formatting the document and formatting the document based on the formatting information. Accordingly, a detailed description thereof is omitted for the sake of brevity.

Once the documents whose URLs are stored in the container have been downloaded, formatted according to the preset formats and configurations, and converted into RTF file(s) in step S1011, in step S1012, the RTF file(s) are output. Alternatively, the RTF file(s) can be edited or saved, depending upon which icon on the graphical user interface has been activated.

The invention has been described with respect to illustrative embodiments. It is to be understood that the invention is not limited to the above described embodiments, and that various changes and modifications may be made by those of ordinary skill in the art without departing from the scope of the appended claims as defined in the appended claims.

APPENDIX 1

SAMPLE USER PROFILE

5 The User Profile is implemented in windows ini file format.

```

10 [Defaults]
    Count=4
    Title=My Daily Paper

    [1]
    Heading=News In Brief
    Site=1
    Section=Front Page
15    MaxLevels=5
    MaxPages=10
    MaxKBytes=2000
    Date=today
    Print=level 0
    Template=1

20    [2]
    Heading=Sports In Brief
    Site=2
    Section=Sports
25    MaxLevels=0
    MaxPages=10
    MaxKBytes=200
    KeywordFilter="Football" AND "49ers"
    Date=today
30    Print=level 0
    Template=1

    [3]
    Heading=Money Matters
    Site=1
35    Section=Business
    MaxLevels=1
    MaxPages=100
    MaxKBytes=2000
    KeywordFilter="Computer" OR "hardware" OR
40    "Software"
    Date=today
    Print=all
    Template=2

45    [4]
    Heading=Sri Lanka
    Site=2
    Section=HotNews
    MaxLevels=1
    MaxPages=100
50    MaxKBytes=2000
    Date=today
    Print=leaves
    Template=2

```

55

SAMPLE SITE PROFILES

```

#Legend:
#%W-day of the week
#%s-section part of URL

[Defaults]
Count=3

[1]
Title=San Jose Mercury News
Username=twickram
Password=cannon
StartData=StartHeadlines
EndData=EndHeadlines
Home Page=http://www.sjmercury.com/
SectionURL=http://www.sjmercury.com/%s.htm
SectionCount=9
Section 1=Front Page
Section 2=International
Section 3=National
Section 4=Local & State
Section 5=Editorials Commentary
Section 6=Business
Section 7=Sports
Section 8=Living
Section 9=Entertainment

[1. Sections]
Front Page=front
International=intl
National=natl
Local & State=loc
Editorials & Commentary=edit
Business=biz
Sports=spts
Living=liv
entertainment=ent

[2]
Title=The San Francisco Chronicle
Home Page=http://www.sfgate.com/chronicle/
SectionURL="http://www.sfgate.com/cgi-
bin/chronicle/article-
list.cgi?S=/chronicle/today"
Section Count=5
Section 1=News
Section 2=Business
Section 3=Sports
Section 4=Editorial
Section 5=Datebook

```

[2.Sections]
 News=News:NN
 Business=Business:BU
 Sports=sports:SP
 Editorial=Editorial:ED
 Datebook=Datebook:DB

[3]
 Title=The Day News
 Home page=
<http://www.lanka.net/lakehouse/ancWeb/dailynew/>
 SectionURL="http://www.lanka.net/lakehouse/ancWeb/dailynew/SW/WS.html"
 SectionCount=12
 Section 1=Business
 Section 2=Editorial
 Section 3=Features
 Section 4=Foreign
 Section 5=Letters
 Section 6=InBrief
 Section 7=HotNews
 Section 8=Probas
 Section 9=Military
 Section 10=Politics
 Section 11=Obituaries
 Section 12=sports

[3.Sections]
 Business=business/intro
 Editorial=editorial/final
 Features=features/intro
 Foreign=foreign/intro
 Letters=letters/final
 InBrief=inbrief/intro
 HotNews=hotnews/intro
 Probas=probas/intro
 Military=military/intro
 Politics=politics/intro
 Obituaries=obituaries/intro
 Sports=sports/intro

APPENDIX 2**SYNTAX FOR RETRIEVAL, EXTRACTION AND PRINTING
CRITERIA**

Maximum levels to search: MaxLevels=<#>

-1: to retrieve all levels

0-n: to retrieve up to n levels

Maximum pages of the document: MaxPages=<#>

n: final document not more than n pages

Maximum size of the document: MaxKBytes=<#>

n: document size not more than n kilo bytes

Exclusion rules:

Date=today|lessthan <#>

today: retrieve only articles posted today

lessthan <#>;n: retrieve only articles no
more than n days old

Retrieve=all|nosubdir|nothisdir|thissiteonly

all: allow to fetch pages from other sites

nosubdir: exclude URLs to subdirectories

nothisdir: exclude URLs in this directory

thissiteonly: fetch pages from this site
only

Keyword search:

KeywordFilter=<keyword> (AND|OR|NOT) <keyword>:

accumulate only pages containing the
combination of keywords

KeywordRank=<#>;n: use fuzzy logic to rank

pages according to keyword combination in

KeywordFilter and keep top n ranked pages

KeywordAuthor=<author>; accumulate only

pages authored by author

ExcludeType=ads|nonEnglish

ads:exclude advertisements

nonEnglish:exclude articles that are not in
English

Flattening rules: Print=all|leaves|level=<#>

all: include all nodes in the tree in the linear
document

leaves: include all leaves in the tree in the
linear document

level=<#>;n: include up to nth level of the tree
in the linear document

Formatting rules: Template=<#>

n: print according to default or user template
number 1

APPENDIX 2

DESCRIPTION OF MODULES

Appendix 3A

THE PERSONAL NEWS PROFILE EDITOR MODULE

The Profile Editor manages access to the user profiles and is represented by CProfileMgr class. It also manages loading and saving of the profiles. The services provided by Profile Editor are:

```

    BOOL NewProfile(CString fileName);
        - Creates a new profile given the
          file name.
    BOOL OpenProfile();
        - Opens the default profile.
    BOOL OpenProfile(CString fileName);
        - Opens the named profile.
    CProfileEntry* GetFirstEntry();
        - Loads and returns the next
          profile entry.
    CProfileEntry* GetNextEntry();
        - Loads and returns the next
          profile entry.
    BOOL WriteEntry(CProfileEntry* entry);
        - Saves a new entry in the profile.
  
```

Each profile entry contains an extraction specification and an output specification as represented by CProfileEntry class. The methods provided are:

```

    CURL GetSiteId();
        - Returns the site id contained in
          the profile entry.
    CExtractionSpec GetExtractionSpec();
        - Returns the extraction
          specification contained in the
          profile entry. Extraction
          specification contains keywords
          for searching, limits for levels,
          pages, size in kilo bytes.
    COutputSpec GetOutputSpec();
        - Returns the output specification
          contained in the profile entry.
          Output specification contains
          formatting instructions and tree
          traversal rules.
  
```

THE Web READER MODULE

5 CWebPage class abstracts the interface to the Internet browser and is representative of the actual Web page. It will be responsible for fetching a Web page, extracting links or references to other URLs in the Web page, and maintaining the contents of a Web page. The methods provided are:

```
10      BOOL Load();
        - Fetch the Web page using the URL,
          username and password.
15      BOOL Parse();
        - Parses the data in the Web page
          and creates a list of links.
          Also resolves the relative URLs
          into absolute URLs.
20      CURLList* GetLinks();
        - Returns the list of links in the
          Web page.
      CPageData* GetData();
        - Returns the actual text data
          contained in the Web page.
25      void FilterContent();
        - Extracts title and other
          information according to the site
          data.
30      CString GetTitle();
        - Returns title and other
          information according to the site
          data.
35      CString GetAnchor();
        - Returns the author of the Web
          page.
      int GetSize();
        - Returns the size of the data in
          clip bytes.
40
```

45 CNetwork class will encapsulate OLE functionality and provides communication with the Internet browser.

```
      CString GetUsername();
        - Returns the currently set
          username.
50      void SetUsername(LPCTSTR);
        - Set the current username in the
          CNetwork object.
55
```

```

CString GetPassword();
    - Determine the currently set
      password.
5 void SetPassword(LPCWSTR);
    - Set the current password in the
      CNetwork object.

void Close();
10 - Disconnect any active connection
    and reset the CNetwork object.

short Read(BSTR* purl, short nAmount);
    - Read data as allowed by the
      Browser.

15 long GetStatus();
    - Query the status of the current
      load.

BOOL Open(LPCWSTR pURL, short nMethod,
LPCWSTR pPostData, long nPostDataSize,
20 LPCWSTR pPostHeaders);
    - Initiates the retrieval of a URL
      from the network.

CString GetErrorMessage();
25 - Provide the caller with
    internally generated error
    messages.

short GetServerStatus();
    - Determine the error status
30 reported by the server.

long GetContentLength();
    - Return the content length (total
      amount or bytes) of the current
      load.
35 CString GetContentEncoding();
    - Return the MIME encoding of the
      current load.

CString GetExpires();
40 - Return when the data retrieved by
    this load is no longer considered
    valid.

CString Receive(LPCWSTR pBase, LPCWSTR
pRelative);
45 - Generate an absolute (fully
    qualified) URL.

BOOL IsFinished();
    - Determine if a load is complete.
50 short BytesReady();
    - Inform the caller of the number
      of bytes prepared to be read.

```

55

THE SITE DRIVER MODULE

The Site Driver will provide the site information to the Web Reader. The Site Driver is functionally similar to the Profile Editor and is represented by CSiteDriver class. Services provided are:

```

    5      BOOL NewProfile(CString fileName);
        - Create a new profile given the
        - file name.
    10     BOOL OpenProfile();
        - Open the default profile.
    15     BOOL OpenProfile(CString fileName);
        - Opens the named profile.
    20     CSiteProfile* GetFirstSite();
        - Loads and returns the first site
        - entry.
    25     CSiteProfile* GetNextSite();
        - Loads and returns the next site
        - entry.
    30     BOOL WriteEntry(CSiteProfile& entry);
        - Saves a new entry in the profile.
    35     int NumberOfSites();
        - Returns the number of sites
        - specified in the profile.
    40
    45
    50
    55

```

An entry in the site profile will contain information about the base URL of the site, title of the news source, information about how to access the site, and various other information such as section data etc. and will be represented by CSiteEntry class. Methods provided are:

```

    35     CString GetURL();
        - Returns the base URL of the site.
    40     CString GetUsername();
        - Returns the username for the
        - site.
    45     CString GetPassword();
        - Returns the password for the
        - site.
    50     CString GetTitle();
        - Returns the title of the news
        - source.
    55     int SectionCount();
        - Returns

```

Appendix 3B**TREE MANAGER MODULE**

5

10

Tree Manager will maintain the most central data structure in this program, which is a tree of Web page nodes and is represented by the CPageTree. CPageTree will traverse the WWW to retrieve the necessary Web pages according to the extraction specification and build the tree. The methods provided are:

15

```
CPageTreeNode* GetRoot();
```

```
- Returns the root node of the tree.
```

```
BOOL Build(CURL URL, CExtractionSpec& spec);
```

20

```
- Builds the tree according to the personal news profile extraction specification.
```

25

Each node in the page tree is represented by a CPageTreeNode. Methods provided are:

```
BOOL AddChild(CWebPage* page);
```

30

```
- Adds a child node with Web page data.
```

```
CWebPage* GetPage();
```

```
- Returns the Web page contained in the node.
```

35

```
int NumberOfChildren();
```

```
- Returns the number of children belonging to the node.
```

```
BOOL IsLeaf();
```

40

```
- Returns TRUE if a leaf node, i.e., no children.
```

45

To traverse the Web page tree, a CTreeIterator class is defined with different traversal methods. Methods provided are:

```
void Reset();
```

50

```
- Cancels the current traversal, and initializes state data.
```

```
CPageTreeNode* GetNextNode();
```

```
- Returns the next node in the tree in a depth first search.
```

55

```

CPageTreeNode* GetNextSibling();
- Returns the next node in the tree
  in a breadth first search.
5 CPageTreeNode* GetNextLeaf();
- Returns the next leaf in the tree
  in a depth first search.

```

10 THE FORMATTER MODULE

Input to this module will be the Web page tree created by the Tree Manager and the output specification contained in the user profile.

15 Formatter will traverse the tree according to the rules specified in the output specification and the final document will be formatted using the formatting instructions in the output specification and the formatting contained in the Web pages such as headings, paragraphs and lists etc.

The output document will be in Rich Text Format (RTF) and will be accessible by many applications. RTF is an advanced formatting language for text, providing document, section and paragraph formatting, style sheets, tables and borders, and with support for Unicode. Image formats supported are DIB, DDB, WMF, OS/2 metafiles. There is no support for Web images which are of the GIF format. A third party library will need to be purchased in order to do the conversion of the GIF to DIB format or one can be developed in-house.

35 The prototype creates a HTML file as the output.

The formatter is represented by the CFormatter class. The methods provided are:

```

40 BOOL OpenHTMLFile(CString fileName);
- Opens the named HTML file for
  output.
void CloseHTMLFile();
- Closes and saves the HTML file.
45 BOOL PrintHTML(CPageTree* root,
  COutputSpec* format);
- Given the root and the output
  specification, traverses the tree
  and prints the contents in the
  Web pages in HTML format.
50 BOOL OpenRTFFile(CString fileName);
- Opens the named RTF file for
  output.

```

```

void CloseRTFFile()
-   Close and sever the RTF file.
5  BOOL PrintRTF(CPageTree& root, COutputSpec&
    format);
-   Given the root and the output
    specification, traverses the tree
    and prints the contents in the
10  Web page in RTF format.

```

```

BOOL Print(CPageTree& root, COutputSpec& format);
-   Given the root and the output specification
15  at, traverses the tree and prints the
    contents on the Web pages to the default
    printer.

```

Claims

1. A method for formatting data from at least one hypermedia document, comprising the steps of:
 - an accessing step to access the at least one hypermedia document;
 - a retrieving step to retrieve data from the hypermedia document into an extracted data tree, wherein the data is retrieved based on a structure of the hypermedia document;
 - a flattening step to flatten the extracted data tree into a flattened document; and
 - a formatting step to format the flattened document into a formatted document.
2. The method of Claim 1, further comprising the step of printing the formatted document.
3. The method of Claim 1, wherein said hypermedia document is located on the World Wide Web.
4. The method of Claim 1, wherein said hypermedia document is located on the Internet.
5. The method of Claim 1, wherein said hypermedia document is located on an intranet.
6. The method of Claim 1, wherein said accessing step, said retrieving step, said flattening step, and said formatting step are performed in accordance with a set of commands.
7. A method of creating a personal news profile, comprising the steps of:
 - accessing the hypermedia-linked computer network;
 - entering a learning mode;
 - traversing sites on the hypermedia-linked computer network with commands;
 - extracting at least one rule from the commands; and
 - compiling the at least one rule into the personal news profile.
8. The method of Claim 7, wherein the at least one rule specifies structural characteristics of sites for traversing the hypermedia-linked computer network.
9. The method of Claim 8, wherein the at least one rule specifies criteria-based criteria for traversing the hypermedia-linked computer network.
10. A personalization system for creating a personal profile for a Web data retrieval system, the per-

sonalization system comprising:

- an input device for inputting data and commands to access the World Wide Web;
- a connection to the World Wide Web;
- 5 a memory for storing a Web reader, the Web reader for accessing the World Wide Web via the connection to the World Wide Web according to commands from the personalization system; and
- a processor for launching the personalization system in response to a user command, wherein the personalization system, upon being launched, (1) launches the Web reader, (2) accesses the World Wide Web via the Web reader, (3) enters a learning mode, (4) sends commands to the Web reader to traverse the World Wide
- 10 Web according to user commands, (5) extracts at least one rule from the user commands, (6) compiles the at least one rule into a personalization profile, and (7) stores the personalization profile.

11. A method for retrieving articles from a hypermedia-linked computer network and for formatting the articles into a personalized newspaper, the method comprising the steps of:

- 15 retrieving a stored personal-news-profile which contains address data for a site on the hypermedia-linked computer network, command data for accessing data from the site, and newspaper layout commands;
- contacting the site based on address data stored in the personal-news-profile;
- 20 downloading articles from the site based on command data stored in the personal-news-profile;
- flattening the articles into a linear document; and
- formatting the linear document into the personalized newspaper according to layout commands stored in the personal-news-profile.

12. The method of Claim 11, further comprising the step of printing the personalized newspaper.

13. The method of Claim 11, wherein said hypermedia-linked computer network is the World Wide Web.

14. The method of Claim 11, wherein said hypermedia-linked computer network is on the Internet.

15. The method of Claim 11, wherein said hypermedia-linked computer network is on an intranet.

16. The method of Claim 11, wherein the command data for accessing data includes data for selecting articles based on a structure of the site.

17. The method of Claim 10, wherein the command data for accessing data also includes data for selecting articles based on a content of the articles.

18. A World Wide Web site data retrieval system for accessing at least one Web site, for retrieving data from the Web site, and for formatting the data into a personalized document, the system comprising:

- 40 an input device for inputting data and commands to access the World Wide Web;
- a memory for storing a Web site data retrieval driver which includes a Web reader, stored Web site address information, stored Web site commands, and stored format information, wherein the memory also includes process steps to connect to a Web site and to issue commands within the connected Web site;
- 45 a connection to the World Wide Web; and
- a processor for launching the Web site data retrieval driver in response to a user inputting a command to access the World Wide Web, wherein the Web site data retrieval driver, upon being launched, (1) launches the Web reader to connect to the World Wide Web via said connection, (2) retrieves the Web site address information and Web site commands, (3) instructs the Web reader to access the Web site based on the Web site address information and Web site commands, (4) downloads Web site data from the Web site based on the Web site
- 50 commands, wherein the data is downloaded with reference to a linked list so as to avoid hypermedia-links that form loops and so as to avoid downloading data which has already been downloaded, (5) stores the Web site data in a linear document, (6) repeats steps 1 through 5 until all addresses in the stored Web site address information have been accessed, and (7) formats the linear document into the personalized document based on the stored format information.

19. The Web site data retrieval system of Claim 18, wherein the Web site address information, the Web site commands, and the format information stored in the memory form a personal-news-profile.

20. The Web site data retrieval system of Claim 13, further comprising a printer for printing the personalized document.
21. The Web site data retrieval system of Claim 16, wherein the personalized document represents a personalized newspaper.
22. The Web site data retrieval system of Claim 18, wherein the personalized document represents a personalized magazine.
23. The Web site data retrieval system of Claim 18, wherein the personalized document represents a personalized book.
24. Computer executable process steps stored on a computer-readable medium, said steps for accessing World Wide Web sites for retrieving data at the sites and for formatting the data into a personalized document, said steps comprising:
- a connecting step to connect to the World Wide Web;
 - a retrieving step to retrieve user-defined Web site address information, user-defined Web site commands, and user-defined formatting commands;
 - an activating step to activate a Web reader so as to access a Web site based on the user-defined Web site address information and retrieving data from within the Web site based on the user-defined Web site commands;
 - a downloading step to download the retrieved Web site data from the accessed web site into an extracted data tree;
 - a flattening step to flatten the extracted data tree into a linear document;
 - a step to repeat the downloading step and the flattening step until all addresses in the user-defined Web site address information have been accessed; and
 - a formatting step to format the stored data into the personalized document based on the user-defined formatting commands.
25. The computer executable process steps of Claim 24, further comprising a spooling step to spool the personalized document to an output device.
26. The computer executable process steps of Claim 25, wherein the output device is a printer.
27. The computer executable process steps of Claim 24, wherein the output device is a display.
28. The computer executable process steps of Claim 24, wherein the user-defined Web site commands include commands for selecting data based on a structure of the Web site.
29. The computer executable process steps of Claim 26, wherein the user-defined Web site commands also include commands for selecting data based on a content of the Web site.
30. An apparatus for retrieving news articles from online news services on the World Wide Web and formatting the news articles into a personalized newspaper, the apparatus comprising:
- first storage means for storing (1) a personal-news-profile, which comprises addresses data and command data for accessing data from a Web site, and (2) newspaper format commands;
 - retrieval means for retrieving the stored personal-news-profile and address data stored therein;
 - activating means for activating a Web reader to contact a Web site based on address data stored in the personal-news-profile;
 - downloading means for downloading news articles from the contacted Web site based on command data stored in the personal-news-profile;
 - second storage means for storing the downloaded news articles; and
 - formatting means for formatting the stored news articles into the personalized newspaper based on the newspaper format commands stored in the personal-news-profile.
31. The apparatus of Claim 30, further comprising spooling means for spooling the personalized newspaper to a printer.

32. A method for formatting data from a hypermedia document into a personalized document, comprising the steps of:

- a location specifying step to specify a location in the hypermedia document;
- a type specifying step to specify the type of the hypermedia document;
- 5 a scope specifying step to specify the scope of data to retrieve from the hypermedia document, wherein the scope is based on a structure of the hypermedia document;
- a format specifying step to specify a format for formatting the data retrieved from the hypermedia document into the personalized document;
- 10 an accessing step to access the hypermedia document found at the location specified in the location specifying step;
- a retrieving step to retrieve data from the hypermedia document accessed in the accessing step, wherein the data is retrieved in accordance with the type specified in the type specifying step and in accordance with the scope specified in the scope specifying step; and
- 15 a formatting step to format the data retrieved in the retrieving step into the personalized document, wherein the data is formatted in accordance with the format specified in the format specifying step.

33. The method of Claim 32, further comprising a printing step to print the personalized document.

34. The method of Claim 32, wherein the location specified in the location specifying step is a filename.

35. The method of Claim 32, wherein the location specified in the location specifying step is a uniform resource locator for the World Wide Web.

36. A method of processing a hypermedia document, comprising the steps of:

- 25 accessing the hypermedia document;
- extracting addresses from the hypermedia document;
- storing the addresses extracted from the hypermedia document in a container in a memory;
- activating a processing function to process data stored in the addresses stored in the container;
- 30 downloading the data stored at the addresses in the container into the memory;
- extracting predetermined data from downloaded data in accordance with predetermined configuration information;
- formatting the predetermined data in accordance with predefined formatting settings to generate a formatted document; and
- 35 processing the formatted document in accordance with the processing function.

37. A method according to Claim 36, further comprising a step of previewing the formatted document prior to processing the formatted document.

38. A method according to Claim 37, further comprising the steps of:

- 40 changing the formatting settings after previewing the formatted document and before processing the formatted document in accordance with the processing function;
- re-activating the processing function; and
- 45 re-formatting the data in accordance with changed formatting settings to generate the formatted document.

39. A method according to Claim 36, wherein the addresses are stored in the container in the order that the addresses are input into the container; and

- 50 wherein the processing function processes the predetermined data in the order that the addresses are stored in the container.

40. A method according to Claim 39, further comprising the step of rearranging the addresses stored in the container by dragging and dropping the addresses within the container.

41. A method according to Claim 39, further comprising a step of pulling the formatting settings and configuration information from graphical user interface.

42. A method according to Claim 41, wherein the graphical user interface includes plural processing icons, one of

which activates the processing function.

43. A method according to Claim 42, wherein the graphical user interface is displayed in plural modes.

44. A method according to Claim 43, wherein the plural modes comprise (1) a fully-functional mode in which the graphical user interface displays formatting fields, processing options, menus and the processing icons, and (2) a minimizing mode in which the graphical user interface displays only the processing icons.

45. A method according to Claim 44, wherein the graphical user interface displayed in the minimizing mode is displayed during browsing the hypermedia document.

46. An apparatus for processing a hypermedia document, comprising:

a Web read which accesses the hypermedia document;
 means for extracting addresses from the hypermedia document;
 a memory including a container which stores the addresses extracted from the hypermedia document;
 a graphical user interface having processing icons which activate at least one processing function to process data stored at the addresses stored in the container; and
 processing means which (1) download the data stored at the addresses stored in the container into the memory, (2) extracts predetermined data from downloaded data in accordance with predefined configuration settings, (3) formats the predetermined data in accordance with predefined formatting settings to generate a formatted document, and (4) processes the formatted document in accordance with the processing function.

47. An apparatus according to Claim 46, further comprising previewing means for previewing the formatted document prior to processing the formatted document.

48. An apparatus according to Claim 46, wherein the addresses are stored in the container in the order that the addresses are input into the container; and
 wherein the processing function processes the predetermined data in the order that the addresses are stored in the container.

49. An apparatus according to Claim 46, further comprising dragging and dropping means for dragging and dropping the addresses listed in the container in order to rearrange the addresses in the container.

50. An apparatus according to Claim 46, further comprising inputting means for inputting the formatting settings and configuration information via a graphical user interface.

51. An apparatus according to Claim 50, wherein the graphical user interface comprises plural processing icons, one of which activates to the processing function.

52. An apparatus according to Claim 51, further comprising display means for displaying the graphical user interface in plural modes.

53. An apparatus according to Claim 52, wherein the plural modes comprise (1) a fully-functional mode in which the graphical user interface displays formatting fields, processing options, menus and the processing icons, and (2) a minimizing mode in which the graphical user interface displays only the processing icons.

54. An apparatus according to Claim 53, wherein the graphical user interface displayed in the minimizing mode is displayed during browsing the hypermedia document.

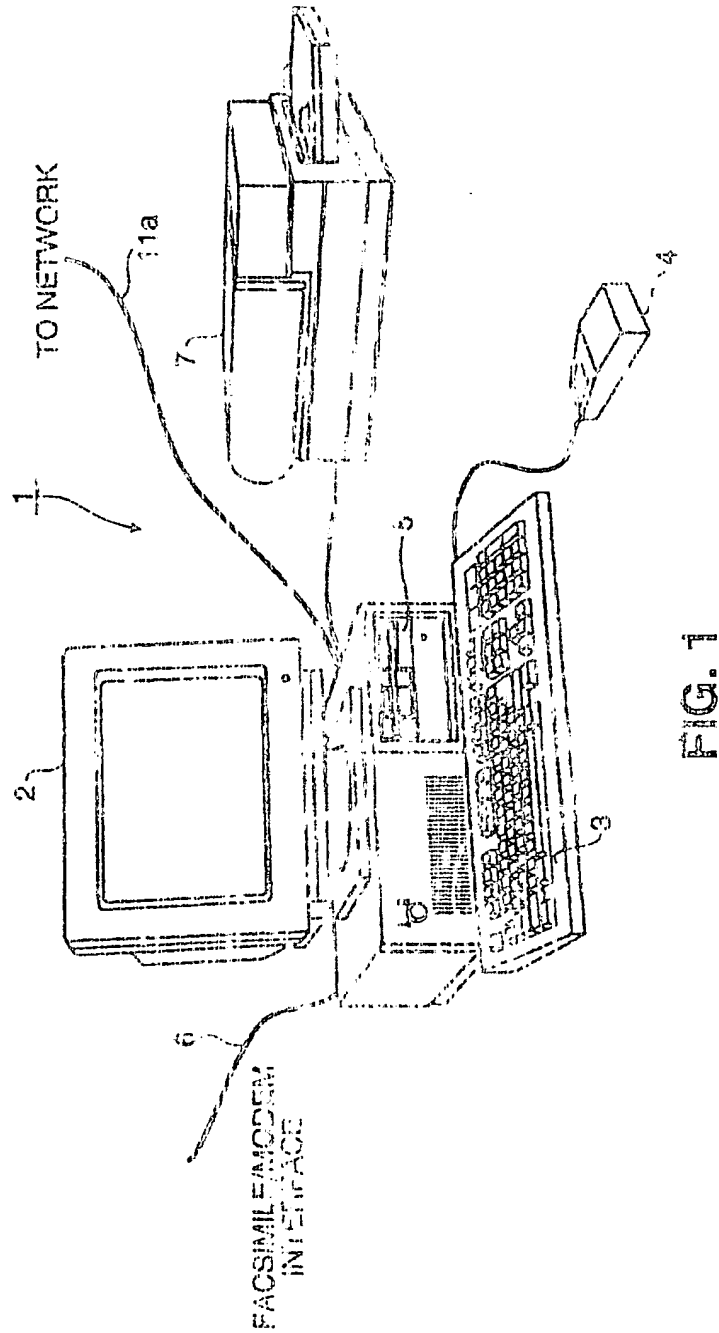


FIG. 1

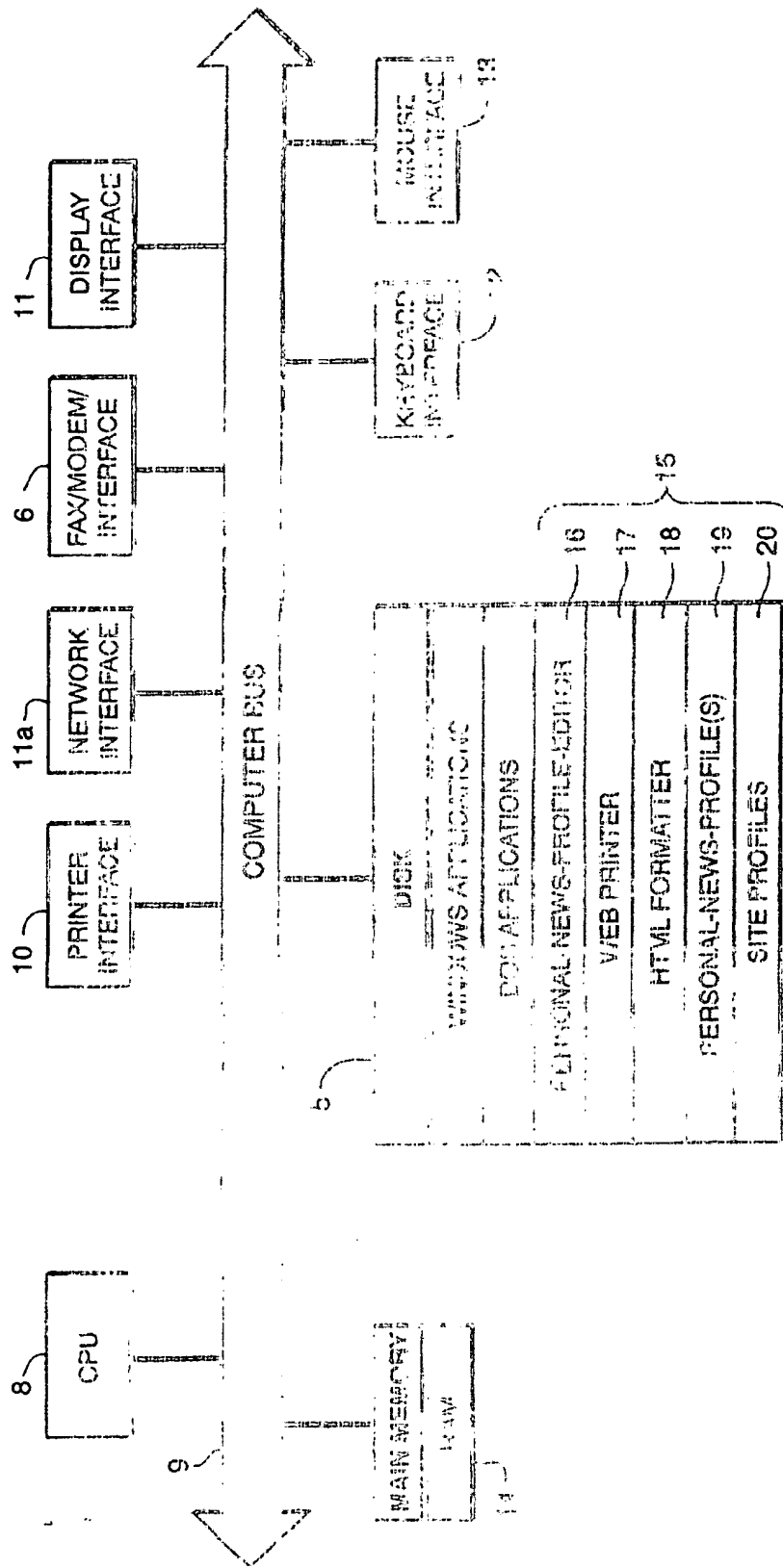


FIG. 2

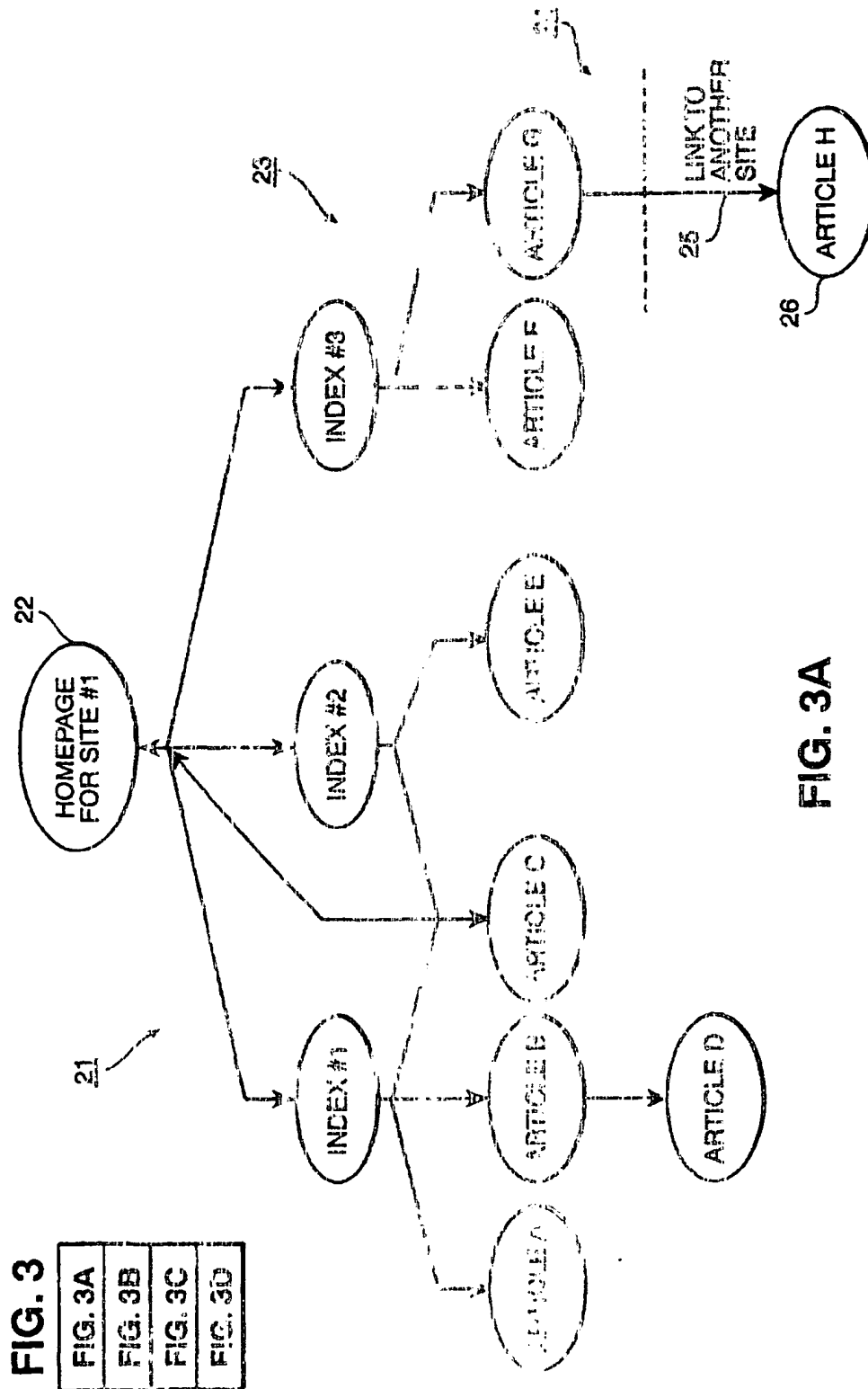


FIG. 3A

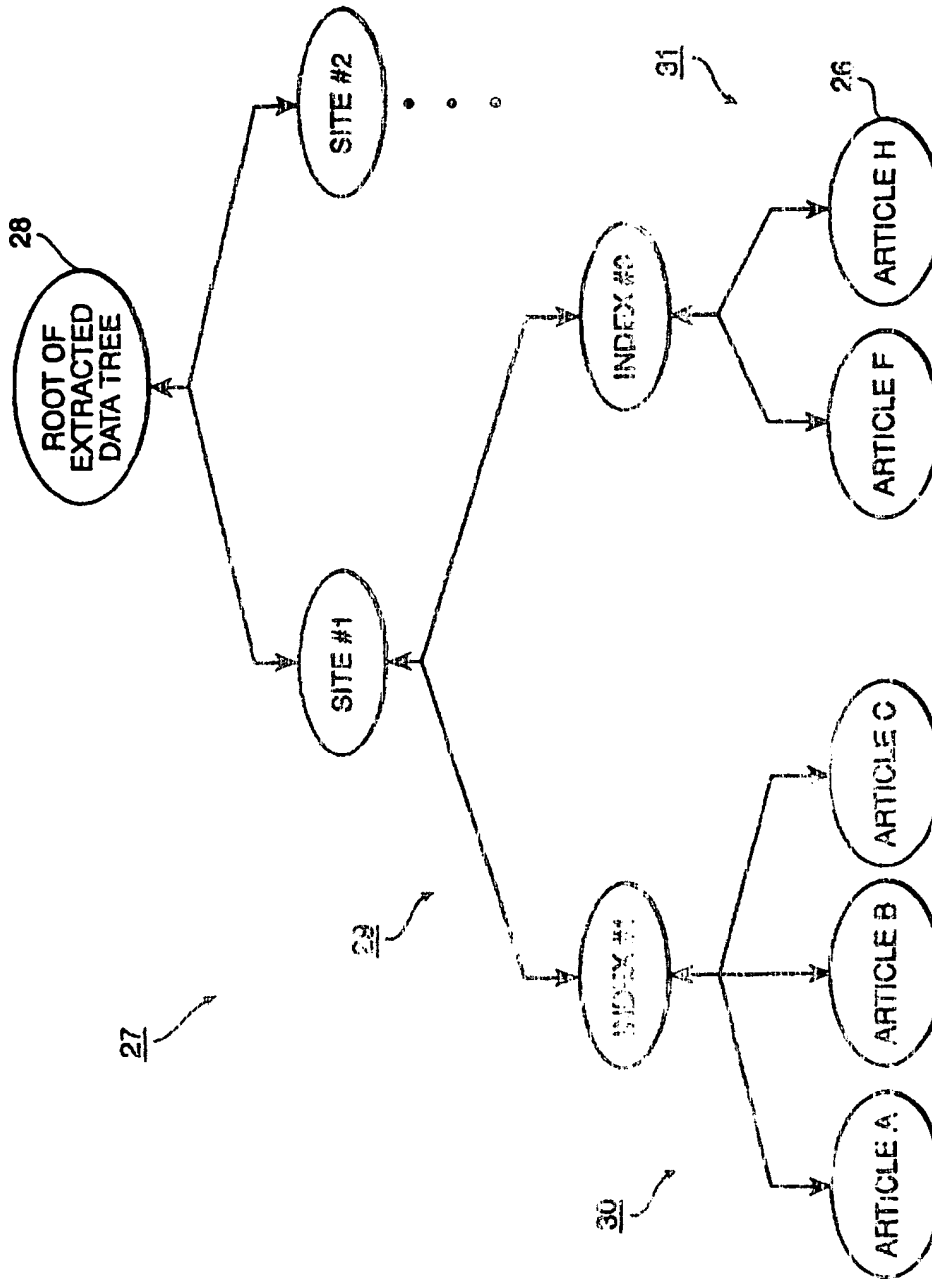


FIG. 3B

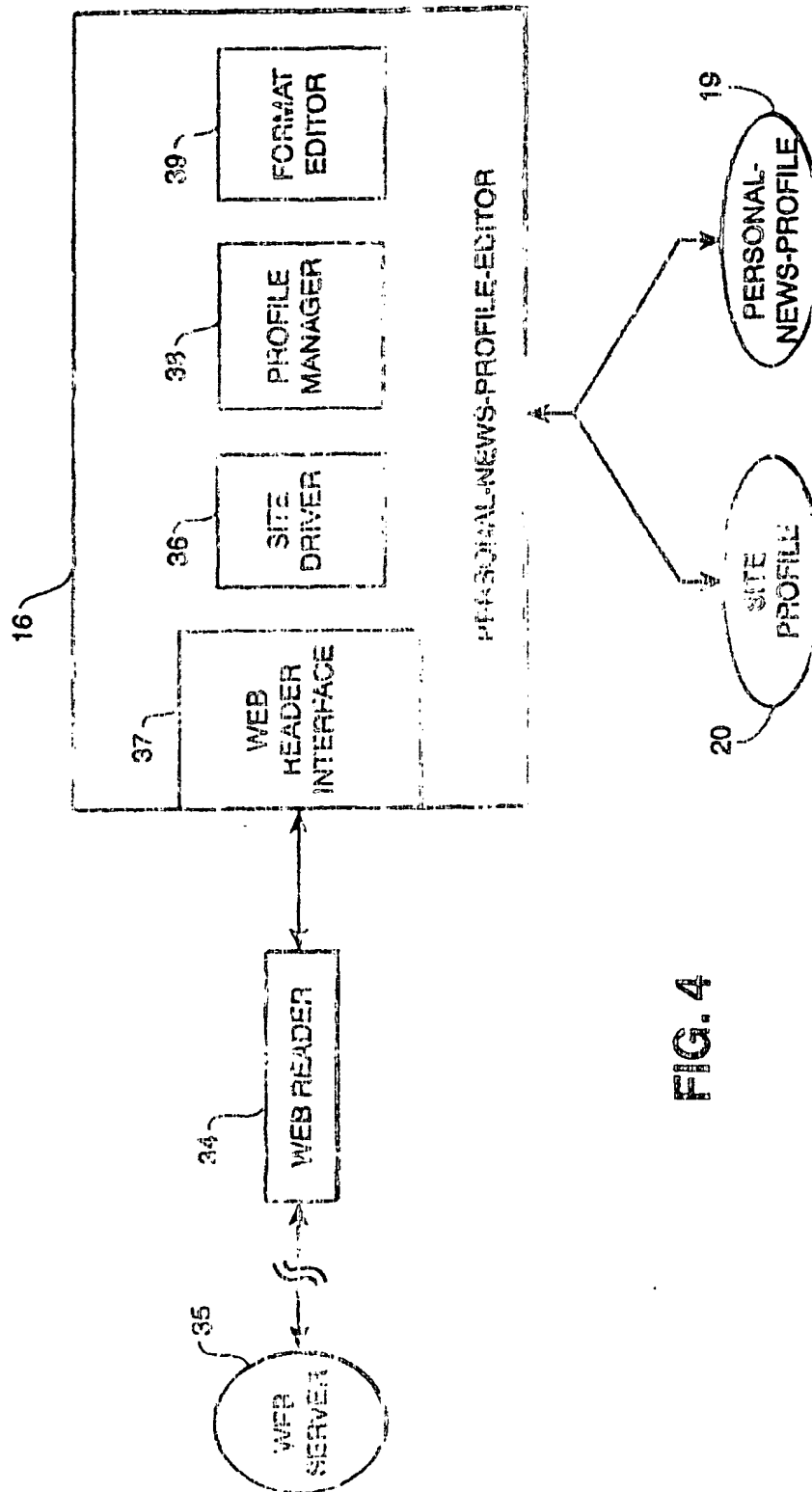


FIG. 4

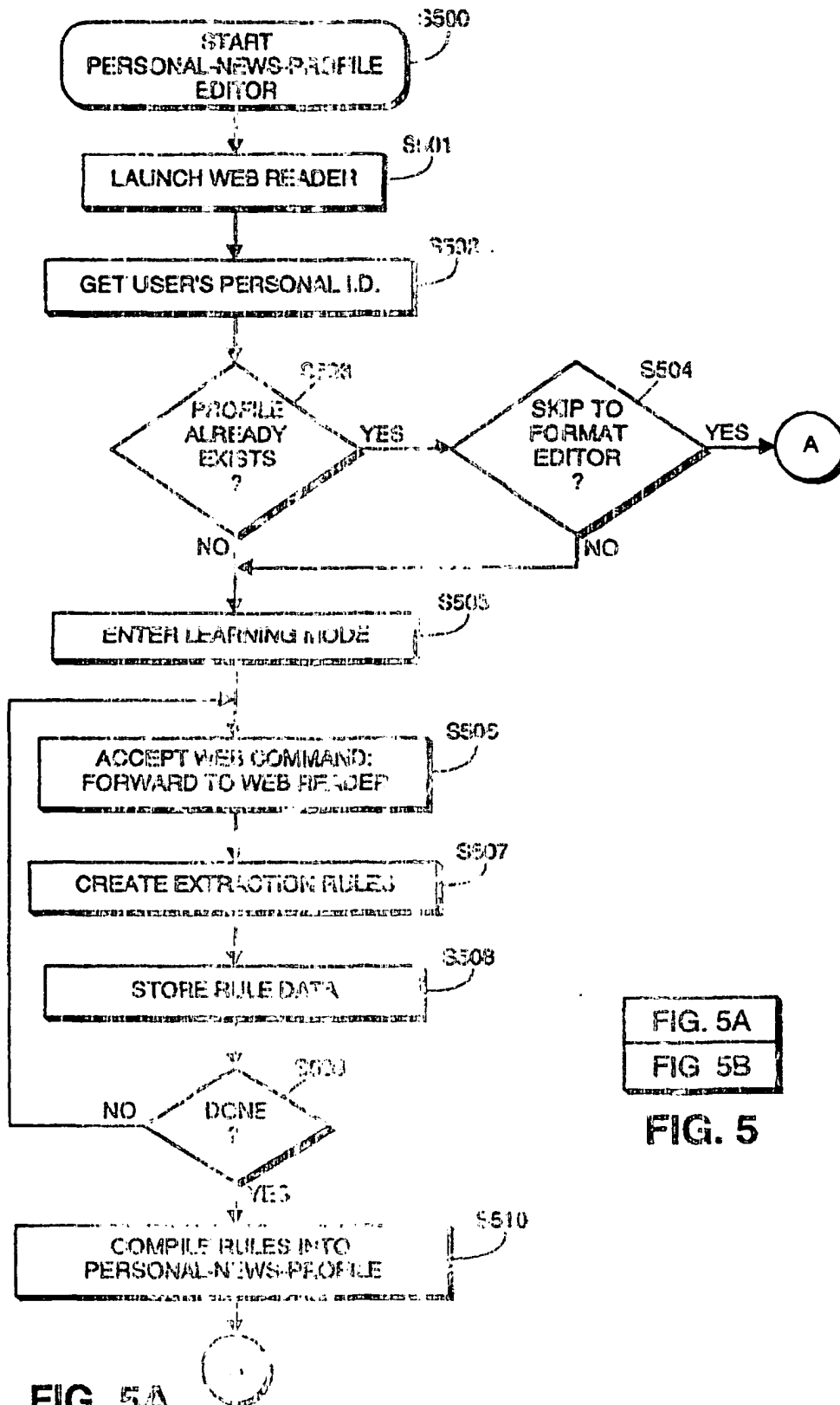


FIG. 5A

FIG 5B

FIG. 5

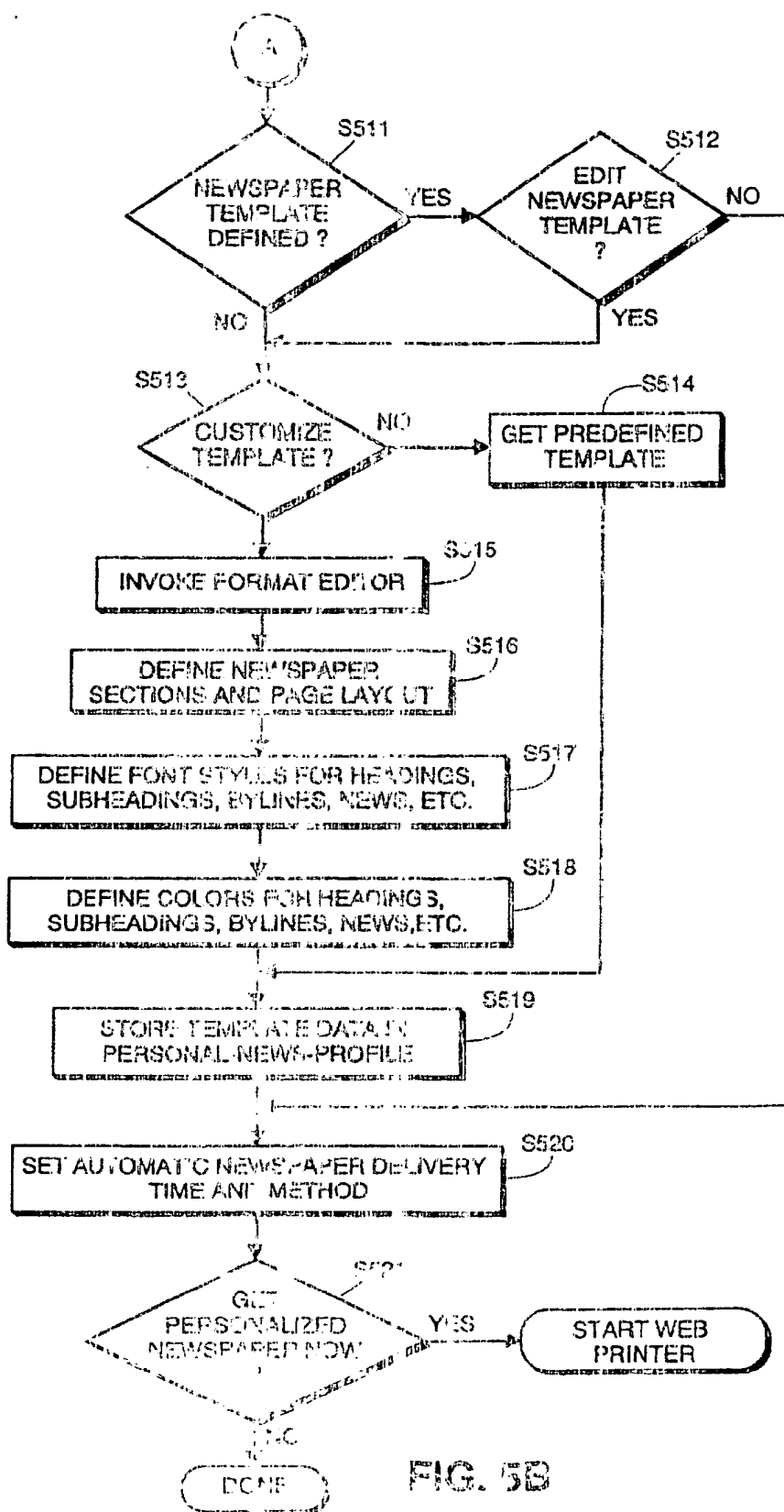


FIG. 5B

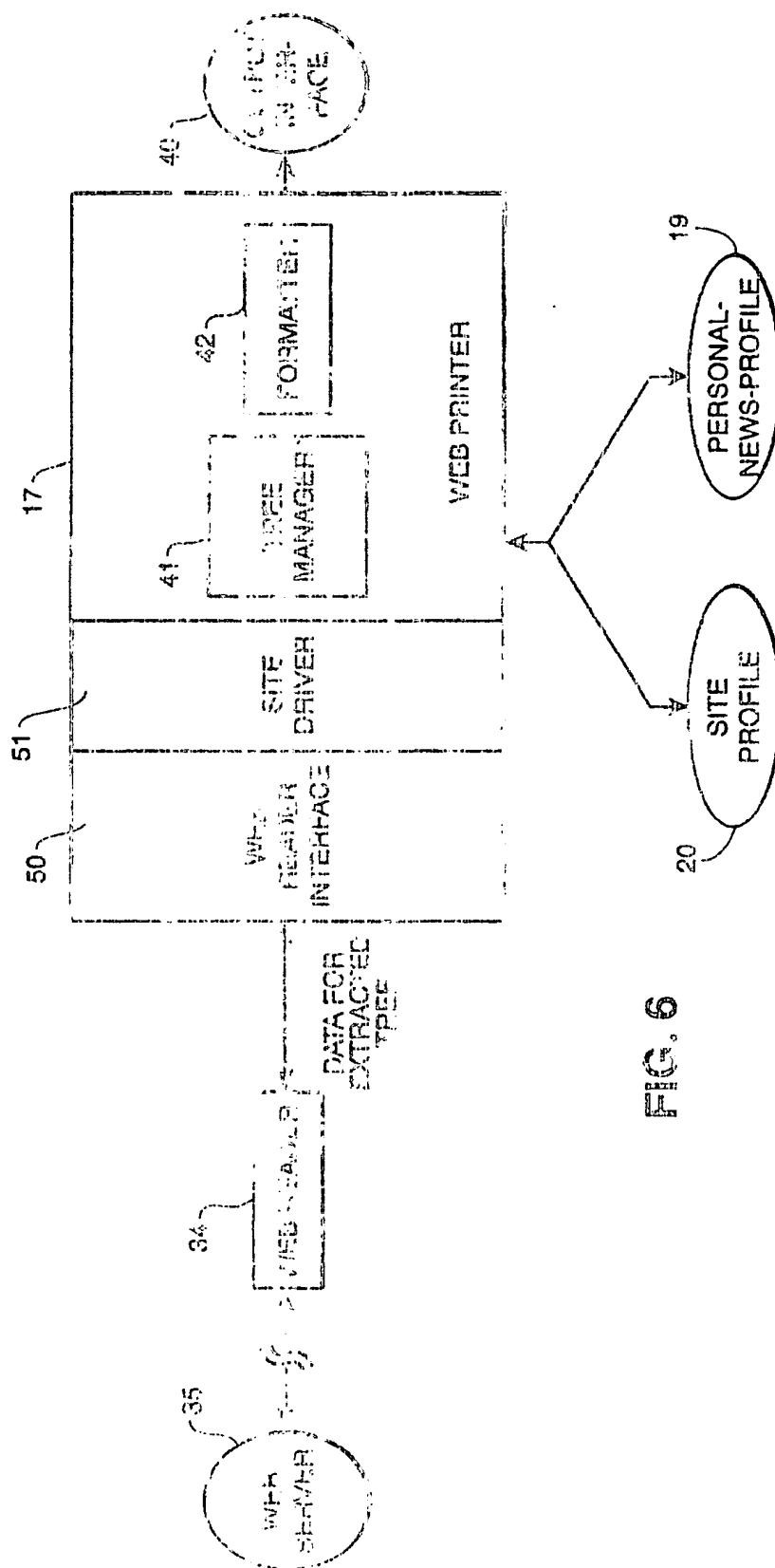


FIG. 6

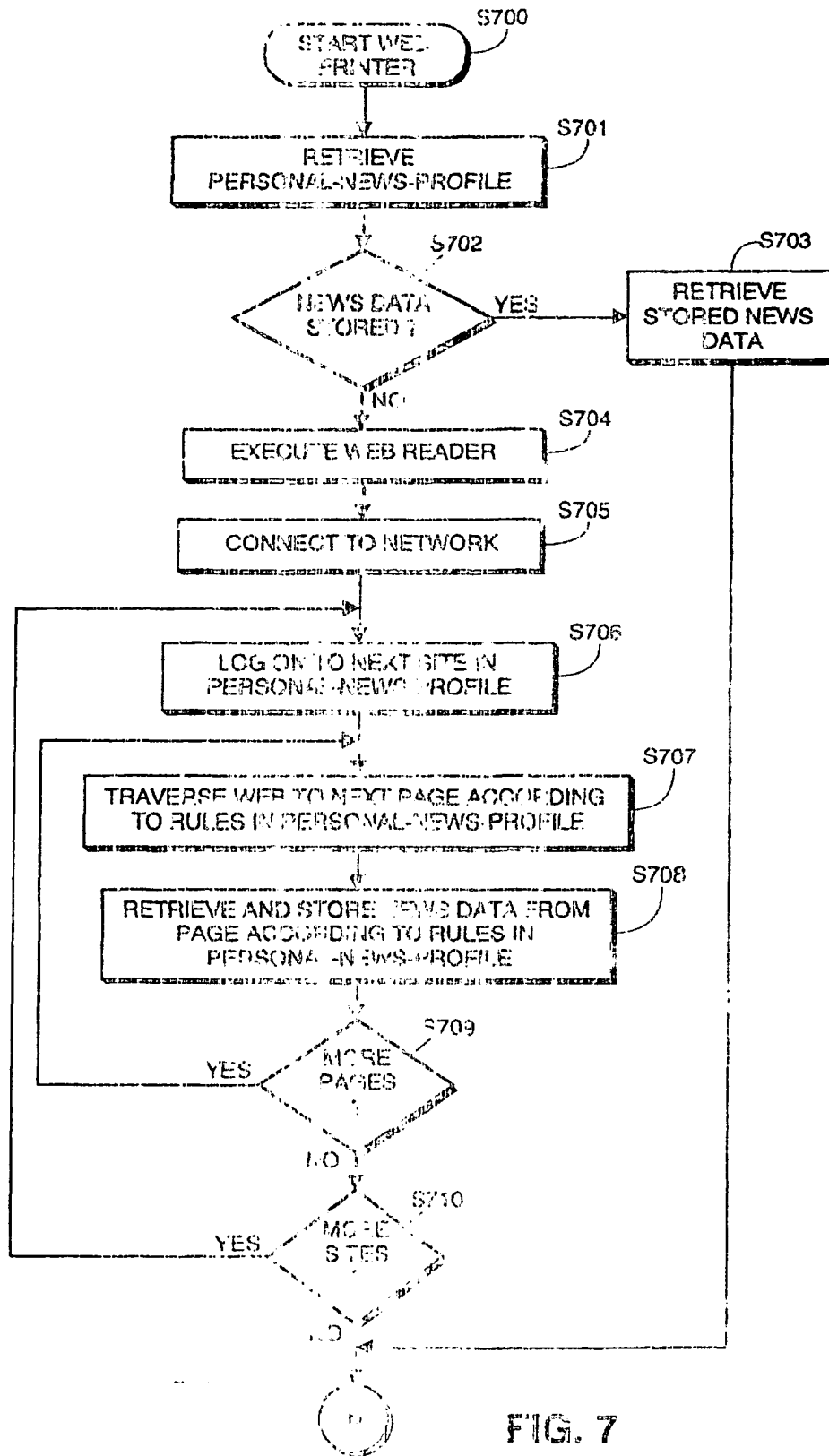


FIG. 7

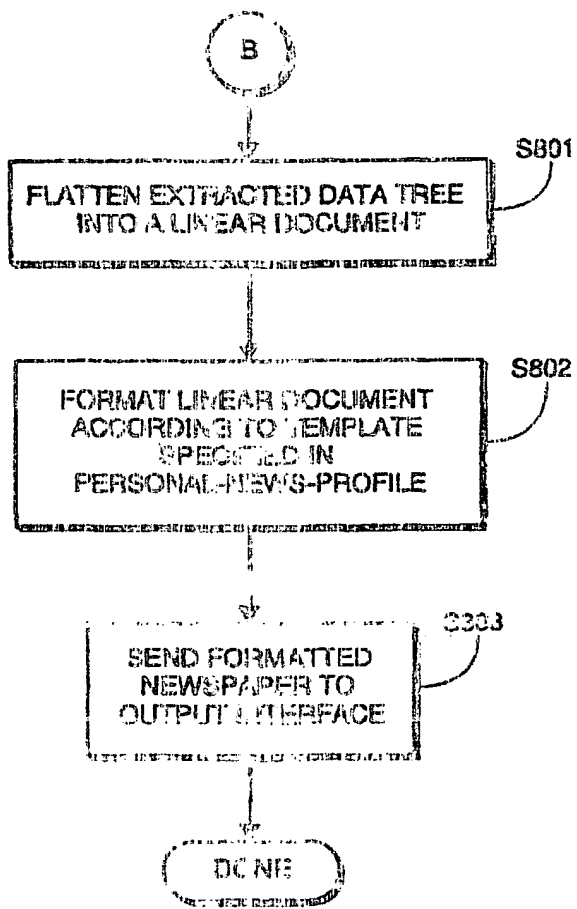


FIG. 3

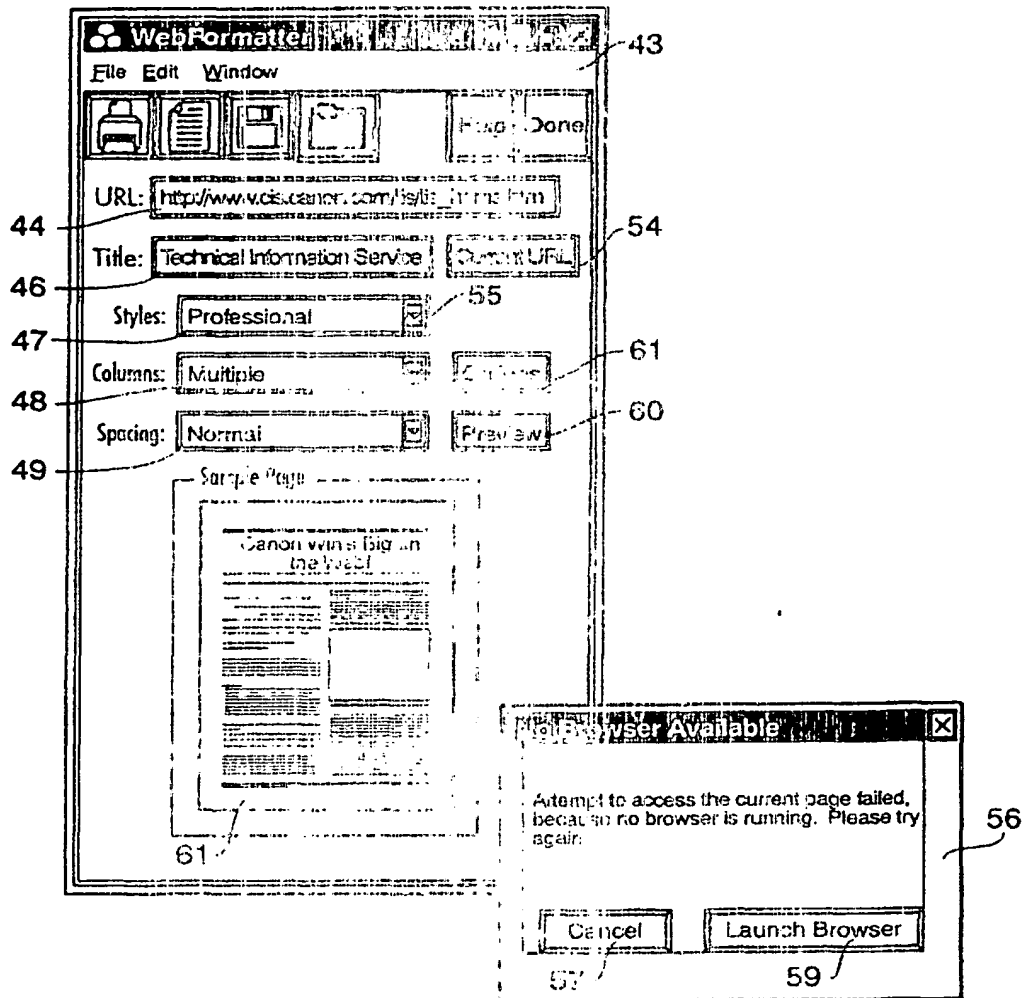


FIG. 1

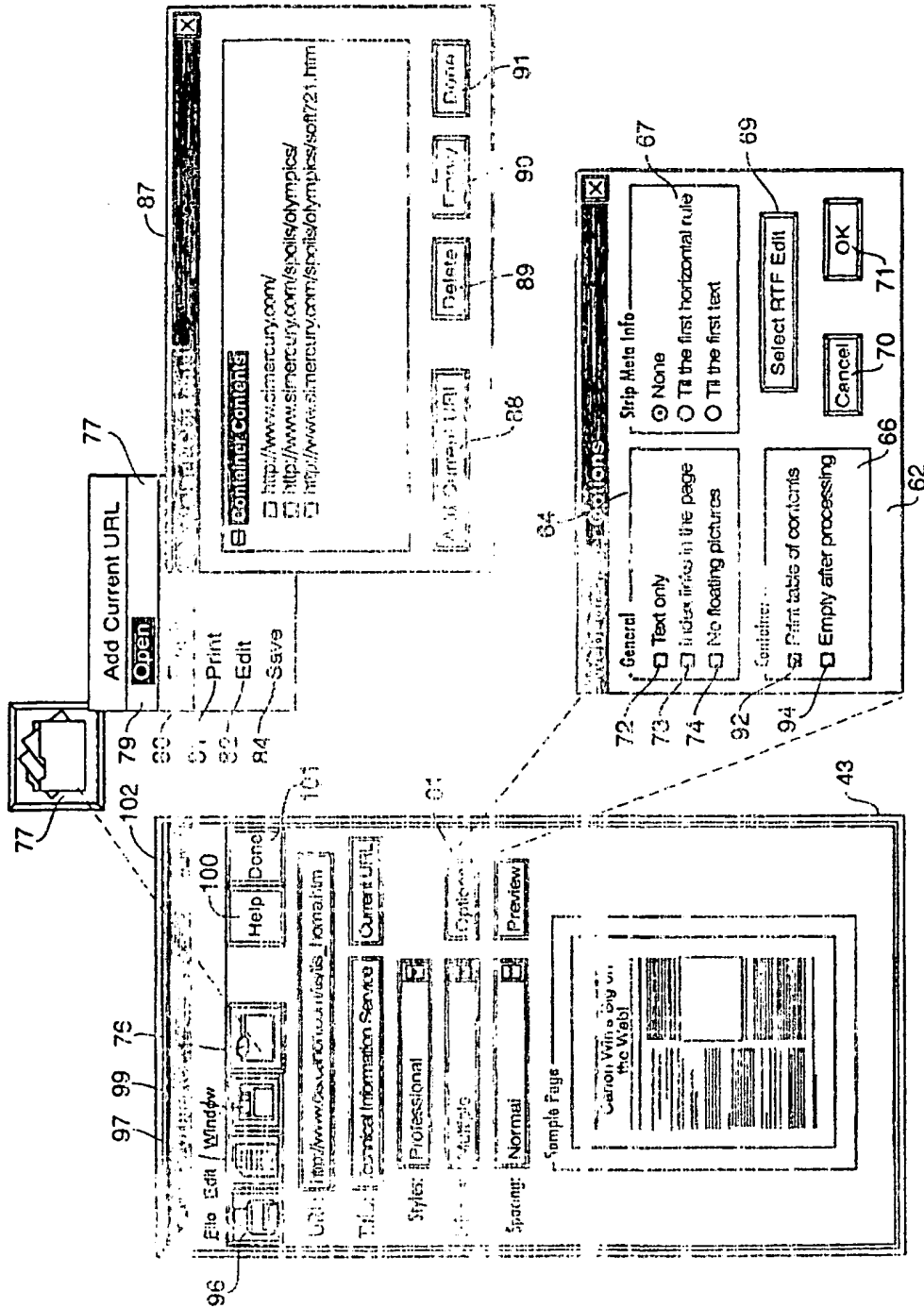


FIG. 9B

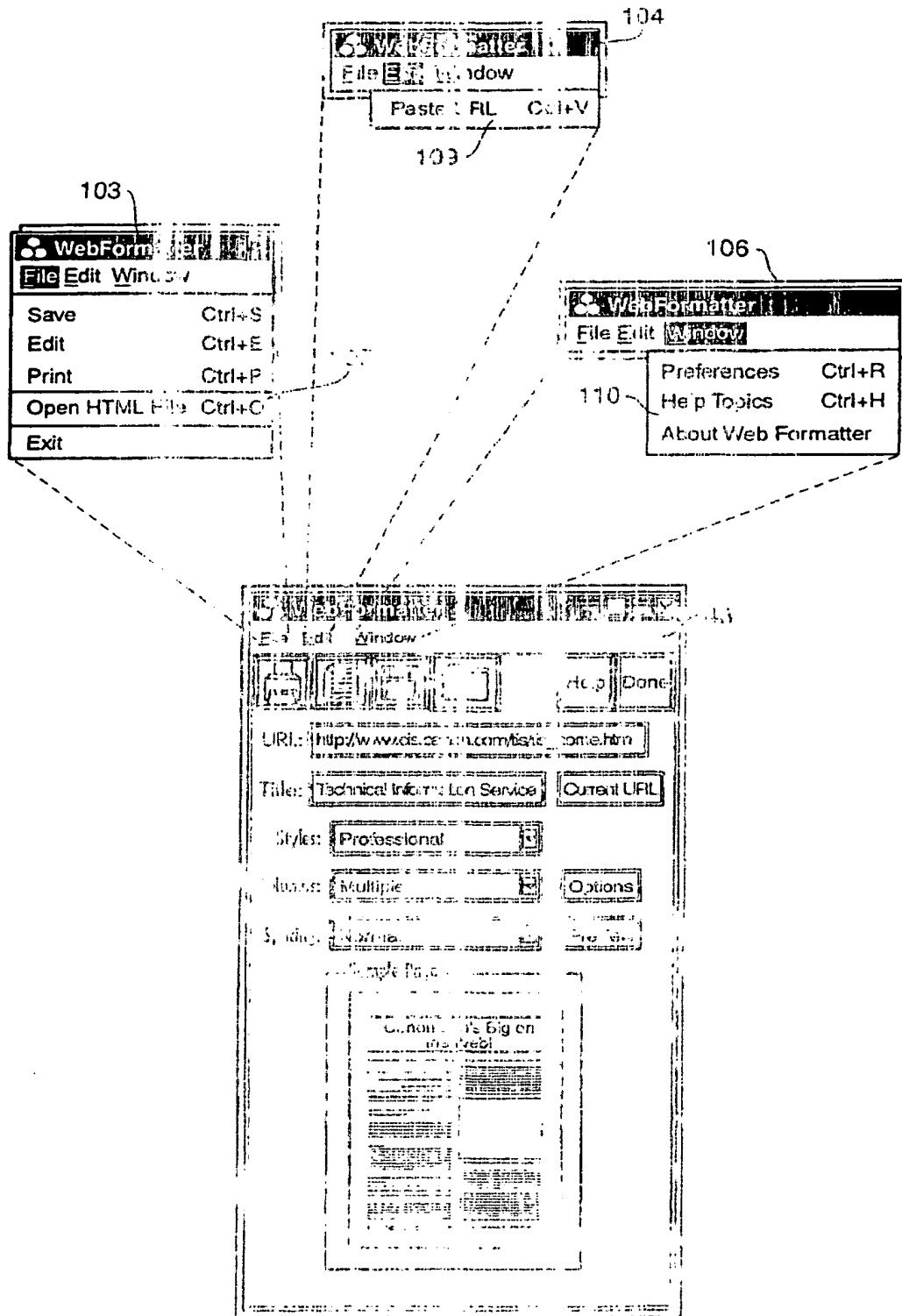


FIG. 9C

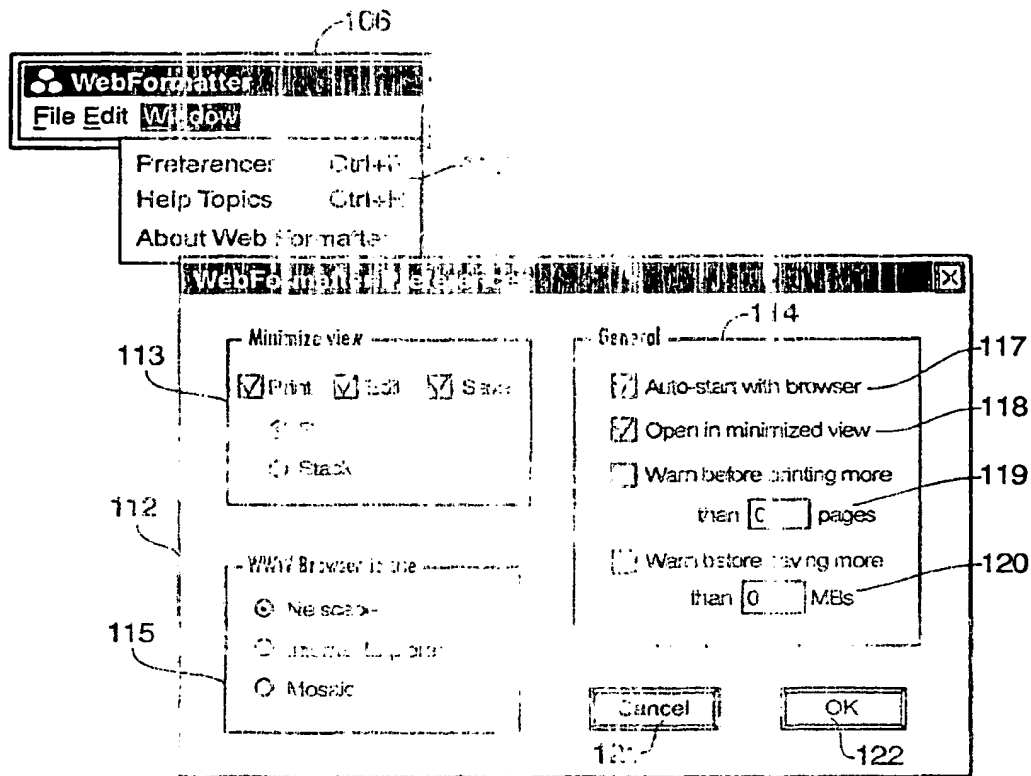


FIG 9D



FIG 9E

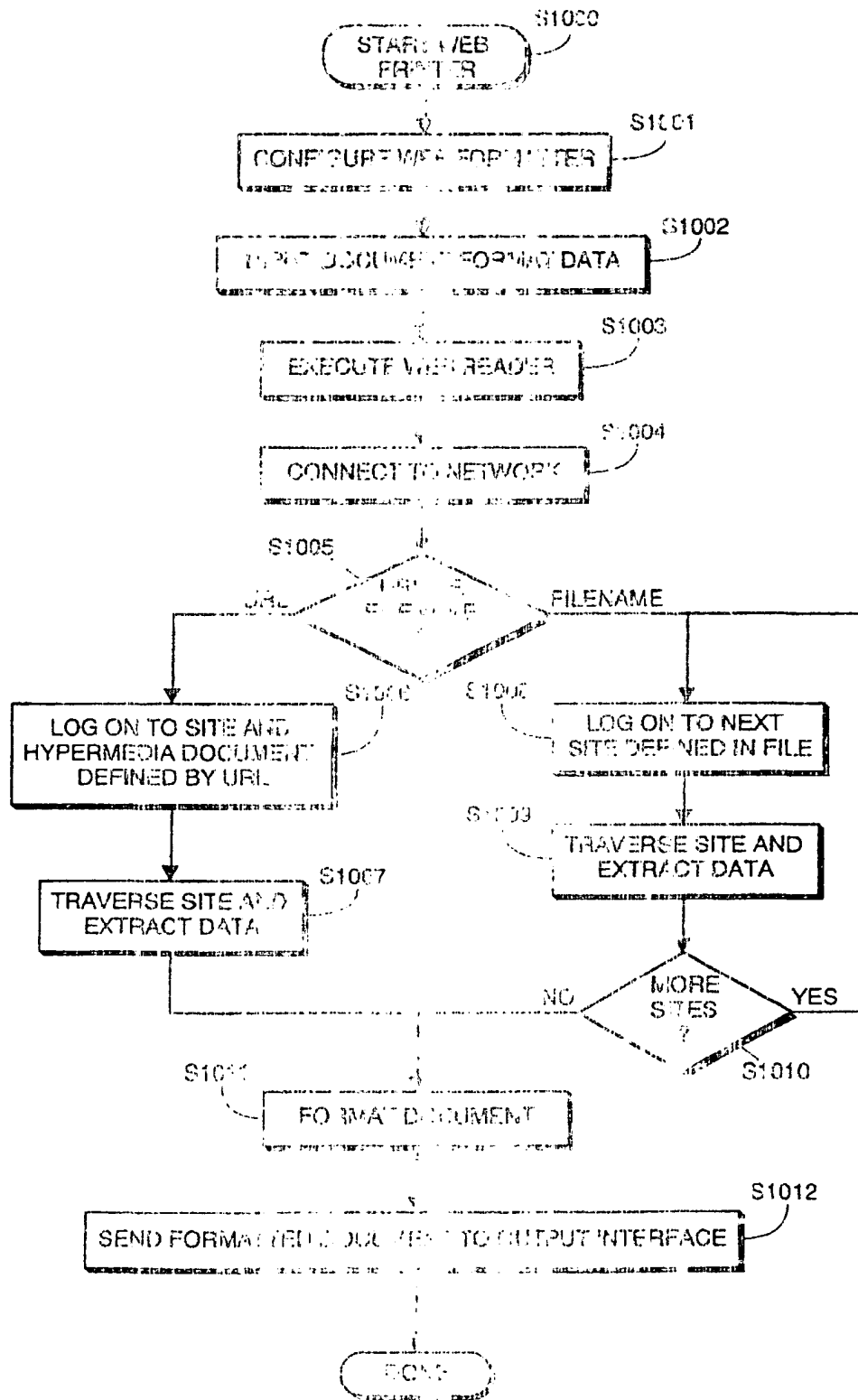
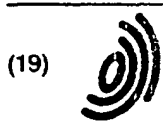


FIG. 10



Europäisches Patentamt
European Patent Office
Office européen des brevets



(11) EP 0 834 822 A3

(12) EUROPEAN PATENT APPLICATION

(88) Date of publication A1:
28.04.1999 Bulletin 1999/17

(51) Int. Cl.⁶ G06F 17/30

(43) Date of publication A2:
08.04.1998 Bulletin 1998/15

(21) Application number: 97307634.7

(22) Date of filing: 10.09.1997

(84) Designated Contracting States:
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC
NL PT SE
Designated Extension States:
AL LT LV RO SI

(72) Inventors:
• Nehab, Srinagar
Costa Mesa, California 92626 (US)
• Wickramaratne, Manjula G.
Costa Mesa, California 92626 (US)
• Mark, Paul L.
Costa Mesa, California 92626 (US)

(30) Priority: 04.10.1996 US 726853

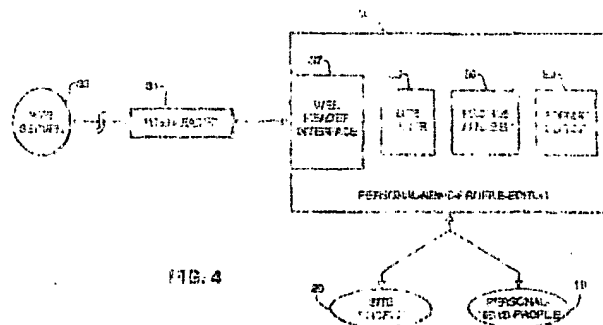
(71) Applicant:
Canon Information Systems, Inc.
Costa Mesa, CA 92626 (US)

(74) Representative:
Beresford, Keith, Denis Lewis et al
BERESFORD & Co.
2-5 Warwick Court
High Holborn
London WC1R 5DU (GB)

(54) World wide web news retrieval system

(57) A World Wide Web site data retrieval system includes an input device for inputting data and commands to access the World Wide Web, and a memory for storing a Web site data retrieval driver which includes a Web reader, stored Web site address information, stored Web site commands, and stored format information. The memory also stores process steps to connect to a Web site and to issue commands within the connected Web site, and a connection to the World Wide Web. The system includes a processor for launching the Web site data retrieval driver in response to a command to access the World Wide Web. The Web site retrieval driver, upon being launched, (1) launches the

Web reader to connect to the World Wide Web via the connection, (2) retrieves the Web site address information and Web site commands, (3) instructs the Web reader to access the Web site based on the Web site address information and Web site commands, (4) downloads Web site data from the Web site based on the Web site commands, (5) stores the Web site data in a linear document, (6) repeats steps 1 through 5 until all addresses in the stored Web site address information have been accessed, and (7) formats the linear document into a personalized document based on the format information.



EP 0 834 822 A3

EUROPEAN SEARCH REPORT

Application Number
EP 97 30 7009



European Patent
Office

EUROPEAN PATENT OFFICE

Application Number
EP 97 30 7009

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
X	<p>LIEBERMAN H: "Letizia: an agent that assists Web browsing" IJCAI-95, PROCEEDINGS OF THE FOURTEENTH INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, PROCEEDINGS OF INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, MONTREAL, QUEBEC, CANADA, 20-25 AUG. 1995, pages 924-929 vol.1, XP002094971 1995, San Mateo, CA, USA, Morgan Kaufmann Publishers, USA; http://lieber.www.media.mit.edu/people/lieber/Lieberary/Letizia/Letizia.html * page 925, left-hand column, paragraph 3 * page 927, left-hand column, paragraph 6</p>	7.10	
			TECHNICAL FIELDS SEARCHED (Int.Cl.6)
The present search report has been drawn up for the reason			
Place of search	Date of classification of the results	Examiner	
THE HAGUE	26 February 1999	Fournier, C	
CATEGORY OF CITED DOCUMENTS		<p>T: theory or principle underlying the invention B: prior patent document, but published in, or after the filing date D: document cited in the application N: document cited for other reasons A: member of the same patent family, corresponding document</p>	
<p>X: particularly relevant prior art documents Y: particularly relevant documents with an abstract or document of the same category A: technological background O: non-written disclosure P: intermediate document</p>			

EPO FORM 1503 03 82 (P04C01)

EP 0 634 622 A3

**ANNEX TO THE EUROPEAN SEARCH REPORT
ON EUROPEAN PATENT APPLICATION NO**

EP 97 30 7009

This annex lists the patent family members relating to the patent document cited in the above-mentioned European search report.
The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

26-02-1999

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0316057 A	24-05-1989	JP 1134561 A	26-05-1989

EPO FORM P0459

For more details about this annex, see Official Journal of the European Patent Office, No. 12/82

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☒ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☒ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☒ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.